MPHIL

Interactive Structural Analysis and Form-Finding

Georgiou, Odysseas

Award date:
2020

Awarding institution:
University of Bath

Interactive Structural Analysis and Form-Finding

by

Odysseas Georgiou

Supervised by

Prof. Paul Richens

Dr. Paul Shepherd

A thesis submitted for the degree of Master of Philosophy

University of Bath

Department of Architecture and Civil Engineering

October 2010

Acknowledgments

I would like to thank my two supervisors, Professor Paul Richens and Dr. Paul Shepherd for their continuous support and for the inspiring conversations which have given me a new insight in the field of computational engineering. Furthermore, I would like to thank Buro Happold and the Happold Trust for their financial support.

Abstract

This thesis re-approaches structural engineering through an interactive perspective by introducing a series of tools that concatenate parametric design with structural analysis, thus achieving interoperability between the architectural shape and its structural performance. Furthermore, this research demonstrates how the design can be realised into an efficient structural form by applying novel techniques of form-finding through the exploitation of the created tools. Two new approaches are developed in this thesis. The first approach uses a 2D truss and a free form surface and combines Parametric Design with Structural Analysis by using computer programming to establish a common interactive framework. The second approach utilises the generated framework to apply techniques of form-finding to structural shapes. The 2D truss is form-found to respond to parametric user defined constraints and an efficient grid structure is applied on a free form surface by following the directions of the principal stresses that occur from its structural analysis. The generated output is applied on a real design project and its structural efficiency is compared with structures created using conventional techniques.

Keywords

Structural engineering, parametric design, interoperability, free form, form-finding, analysis, interactive, digital architectonics

Table of contents

List of figures

1 Introduction

The digital age has introduced a new approach to architectural form. The rapid advance in CAD technology has enabled architects to overcome the traditional design boundaries and to transform any imagined shape into a persuasive building. In this context, structural design is lagging behind and engineering engagement with architecture is restricted to making the building stand up. This traditional approach cannot keep up with the modern design process and the engineer is unable to give feedback to the architect's design, often stalling the design process. While a large variety of tools serving architectural geometry, such as parametric modelling, is available for use by architects, allowing limitless capabilities and speed in design, the engineering industry remains adherent to traditional structural analysis techniques.

When it comes to simple engineering problems and buildings that consist of straightforward structural elements such as columns, beams and slabs, the behaviour of which is easily understood and assessed, the engineer can quickly give direct feedback to the architect by using traditional rules of thumb as for example span to depth ratios or section strength tables. The same feedback cannot be given on free form shapes. According to Wagner and Bogle, "*One of the principal problems was related to the difficulty of gaining insight in the structural action of complex three-dimensional layouts, when precedents are few and rules of thumb are not yet established* "[1]. The more complex an architectural shape becomes, the more difficult it is for the engineer to interpret it and understand its structural behaviour. A time consuming process of modelling and simulation, prior to the determination of the structural outcome unavoidably causes friction, which often limits the efficiency of structural design. In order for a model to be shared among the different design disciplines, a large amount of data needs to be exchanged and a number of translations need to be made, an arduous procedure which does not necessarily lead to precise results. This raises the need for better interoperability among the software used by the design team.

By employing the power of parametric design combined with structural analysis software, one can speed up this process and can have the ability to explore more iterations of a structural solution which can lead to the optimum result without being restricted by time barriers. By extending the capabilities of parametric design to include and implement structural analysis, the engineer can move away from the traditional ways of structural thinking and relax the technical boundaries. The results deriving from a structural analysis need no longer be single solutions to problems but parameters that feed into the architectural form and conclude to an optimum shape.

This thesis aims to introduce a series of tools to fill this gap between architectural and engineering design and to enhance the dialogue between their representatives, by combining software tools that are already available to both disciplines, in order to

interactively manage and form-find structures for free form surfaces. These means enable the design engineer to retain better control over software packages for structural analysis and also enhances their capabilities in parametric modelling, interactive results output and efficient form-finding. In addition, by utilizing the combined power of parametric modelling and programming, novel efficient forms of structure are explored and exploited.

In Chapter 3, simple examples of structures are described, leading primarily to full parametric control of their design attributes and direct representation of their analysis results. Secondly, efficient forms that correspond to each possible change of the initial geometry are generated. The case of a two-dimensional, simply supported truss is investigated first, as an initial approach to achieve interoperability in structural design by combining computer programming and parametric modelling. The same example is then used as a starting point to generate more efficient forms of trusses, which also respond to the designer's change of objective.

In Chapter 4, a more complex case is explored, that of a free form surface. A software tool in the context of free form design is developed, to control any type of complex surface, analyse and extract real time results and perform operations within the environment of a parametric design software. By further exploiting this tool, a novel technique for the generation of a structure to support the previously assessed surface is investigated. The technique involves transforming a continuum surface into an efficient grid shell, by following the directions of the principal stresses that occur from the structural analysis of the former. An intermediate study on the analogous case of following principal curvature direction fields is also conducted to prove the feasibility of the method.

Finally, in Chapter 5, the grid generated by the above technique is tested on a real design project and its performance is compared to conventional types of structures that could support free form shapes. Throughout the process, the notion of interactivity and parametric control is maintained, so that the user can determine and manipulate changes at any stage of the design.

2 Literature Review

This literature review documents the basic notions, techniques and recent developments regarding the design of modern architectural forms. The concept of free form architecture is described first by illustrating realised examples of built projects with some of the common tools aiding such designs. Typical structural forms suitable for supporting such designs are presented through their historical development. An attempt is then made to concisely approach the terms "parametric design" and "interoperability" which are widely used throughout this thesis.

2.1 Free form Architecture

The advent of technology and its application to architecture has led to an interest in the development of free form architecture. Free form architecture can be defined as irregularly shaped design, consisting of "doubly curved" surfaces (curved in both directions) which do not particularly feature a component repetition. The distinction between a free form and a form-found shape should be made at this point, as form-found surfaces or shapes are the ones that are optimised to meet structural constraints and thus cannot be defined as free. Free shapes, initially introduced in the car and aerospace industry and later popularised in the film industry, evolved through the advancement of CAD tools [1]. Consequently, these have led architects to start experimenting with new formal expressions, pushing the boundaries to non-conventional designs. Initially, the results of these experiments were visually stunning blobs or liquid shapes rather than orthogonal buildings, that prevailed in the construction industry until then, where distinct structural elements (beam, slabs and columns) could be distinguished.

The Guggenheim Museum, Bilbao by Frank O. Gehry and Associates, completed in 1997 (see figure 1) is a famous example of free form architecture, well known for its irregularly curved shape. Opposing the complex shaped facade, the internal armature of Gehry's building consists of conventional straight steel members connected at nodes, hidden behind the metal clad sheets.

Figure 1- Guggenheim Museum, Bilbao by Frank O. Gehry

Figure 2 - Doha Villa by Ushida Findlay Architects

Another case of a building, as shown in Figure 2, that can be indulgently referred to as a "blob" is Doha Villa (2002) by Ushida Findlay Architects. The structure follows the designed shape by forming reinforced concrete planar ribs around the perimeter and by using infill panels to clad the surface. The ribs' formwork was constructed using CNC (computer numerical control) timber cutting techniques. The infill panels were formed by manually bending steel reinforcement bars between the ribs, creating a surface to be sprayed in concrete.

The project 279 Selfridges by Future Systems, completed in 2003 (see Figure 3), is another realised example of this new form of architecture, where the "fluidity of shape recalls the fall of fabric or the soft lines of a body" [2]. The curved geometry structure was achieved by making use of sprayed concrete technology. A similar construction approach as with Doha Villa was employed: the edges of the structure's slabs where used as guides, instead of ribs in the former case, and a metal mesh was bent on site between them, held in position by adjustable scaffolding, to form the permanent formwork.

Figure 3 - 279 Selfridges project by Future Systems

2.2 Tools used for free form design

A variety of CAD tools are available aiding the modelling of free form surfaces. Amongst the more popular being McNeel's Rhinoceros 3D [3]. Rhino 3D is based on NURBS modelling [4] and extends its use to Parametric Modelling through the Grasshopper 3D's plug-in [5]. The combination of these two packages will be used throughout the thesis research. Autodesk's MAYA 3D [6], used mainly in the 3D animation industry, which enables parametric modelling through its Construction History function, Bentley's Generative Components [7] and Dassault CATIA [8] , mainly used in automotive industry, are other popular software packages that use NURBS engines to shape and parametrically control free form surfaces. All have found some use in architecture.

2.3 Structural Forms

A structural form is one designed to maintain its shape and stability in the presence of loads arising from its own weight and superimposed loads such as those arising from traffic, wind,

water and soil pressure. For any given free form surface structural engineering aims to formulate a solution that will integrate with the architectural shape and will provide stability under external conditions. Obviously, certain constraints within limits of safety and economy should be taken into account in order to make the structure stand in a feasible way. Therefore, it is the job of the structural engineer, in the early stages of design, to advise the architect on viable structural solutions and at a later stage to rigorously design and test the proposed solution. Some of the most common structural solutions to support free form designs are concrete shells, lattice domes, grid shells and slung membranes. These structural forms are not particularly novel in the construction industry; in fact, they prevailed from the middle of the 20th century until they were supplanted by the simplest of all elemental forms: beams, slabs and columns. Concrete shells are common in dome-like structures since concrete is most effective when it acts in compression. Historically concrete shells date back to the 2nd Century, a spectacular survival being the Pantheon in Rome [9]. Two brilliant representatives of modern solid concrete shell engineering are Eduardo Torroja [10] and Felix Candela [11]. A variation of this structural form is Ribbed Concrete Shells comprising of a thin concrete topping and reinforced by deeper beams forming a more lightweight dome-structure. Some great examples of this type of structures can be found by Pier Luigi Nervi [12].

Due to the rising cost of labour and unconventional formwork needed to construct concrete shells, a tendency towards purely tensile structures and space frames was generated and by 1970, the first type of such construction was supplanted. Timber grid-shells are another form of compression structures worth mentioning. They consist of timber lath lattices, flexibly connected at nodes, bent to form a shell-like structure (see Figure 4). After the deformation of the timber lattice, the shape is maintained by fixing the connections in place. Stresses are carried by the individual laths, which act as arches and thus resist mainly compression rather than bending, particularly when shaped in funicular forms. [13]

Figure 4 - Mannheim Multihall, Frei Otto, 1971 (Image courtesy of www.fourthdoor.org)

Tensile Structures, or Pre-stressed Membranes, have evolved through widespread experience in sails, by combining high strength fabric, cables and struts to form large-span doubly-curved surfaces, through complex construction methods that needed higher skilled labour than their prior counterparts did. Some great examples of such structures were designed by Frei Otto and Ted Happold, "the most adventurous pioneer designers of such roofs" [13].

The most recent development of the aforementioned forms, are space-frames, developed with an appreciation of three-dimensional equilibrium and the enhancement of material tensile strength. These can be seen in trussed, or triangulated layouts, and to quadrilateral rigid-jointed grids. The latter form is further exploited during this thesis, aiding the development of novel grid structures.

2.4 Parametric Design in Structural Engineering

The development of complex forms in architecture led to the transition from the traditional, static design approach, to a more "dynamic" process that enables the control of the "inevitable changes" that derive from the former. According to Robert Aish and Robert Woodbury, this differentiation in design methods is described as the following: "Conventional *CAD systems focus design attention on the representation of the artefact being designed. Currently industry attention is on systems in which a designed artefact is represented parametrically, that is, the representation admits rapid change of design dimensions and structure*" [14]. The term "parametric design" has evolved in the last decades through the application of advanced computing to architecture and it refers to designing an object by describing the relationship between its parameters. This allows the re-definition of the final output by manipulating the components of the object. An object is described by its geometrical constraints or by its degrees of freedom as well as by its dimensions and the equations that define their correlation [15]. What makes a difference to its application to architecture, is the association of a whole set of parameters that define new functions in order to fully describe and control a design. In the context of a building, the control of such a set of parameters can enable a holistic, associated approach to architectural design. With this tool provided, the designer is able to describe mathematically complex forms and to discover new ones, while understanding their conceptual functions. Consequently, by overcoming the constraint of time in facilitating changes they can yield them to more optimum solutions. Amongst the most widely used parametric design tools is Bentley's Generative Components, which according to the software research director, Robert Aish "*has the potential to span the architectural process from concept formation to digital fabrication in a system of related design models"* [16]. A debatable drawback to the multiple virtues of the application of parametric design in architecture is that it increases complexity and without the proper structural guidance from the early design stages, often leading to results that are difficult to construct.

While a variety of parametric design software (see section 2.2) is readily available to fit the context of architectural design or more specifically to analyse and manipulate geometry, they are not widely applied to structural engineering problems. According to the description of Generative Components, "*it consists of a rich set of predefined geometric types and relationships*..." and *"...includes predefined measurement tools which can be used to evaluate the performance of such geometric construction."* [16] Such extensions, serving geometry and consequently architectural design, are not available, at least commercially, in a form relevant to the work of a structural engineer. Since building forms are becoming more complex, the engineering input is becoming essential from the concept stages and the role of the engineer in a design collaboration is enhanced. Despite that, there seems to be a gap between the ways the two disciplines interact. While a concept form can be shaped, understood and altered quickly by the architect, an engineer cannot assess the date given in analogous speed. For a given free form shape, developed with ease inside a Parametric CAD environment, extensive analysis needs to take place on the engineering side for it to be structurally assessed. Most of the times, the full behaviour is not fully understood due to the time chasm between modelling, analysis and assessment and just static performance feedback is returned for the shape analysed. Structural constraints, such as performance of structural members or materials and boundary conditions for example, are not controlled equally by Structural Analysis software as opposed to the way geometrical constraints are treated by parametric design software. Conceptual structural engineering of complex shapes is lagging behind the architecture that it is summoned to support. While engineers are already using rules of thumb to quickly assess simple orthogonal structural problems, for example how deep a concrete beam has to be for a given span, or the type of section suitable for a compression member, there are not any tools available to give a quick feedback or even understand the performance of a free form surface.

The engineering industry is currently making limited use of parametric CAD tools, as far as the extents of structural modelling are concerned. Only a few structural engineering practices, as Buro Happold (S.M.A.R.T. Team) and Adams Kara Taylor (Parametric Applied Research Team, "p.art"), are developing in-house software to enhance the design process. These tools are still only dedicated to the specialized teams that create them rather than used by the mass of practitioners.

A successful attempt to enable a performance based generative design is documented by Kristina Shea [35] in developing a synergy between a structural design system and Bentley's Generative Components using XML models.

2.5 Interoperability in design

Since the Renaissance, engineering and architecture have been forced to diverge to different disciplines due to a demand of more rigorous design and in order to serve a sustainable and structurally safe built environment. Following the initial split, this chasm

has become larger as each discipline tends to specialise more in their new "dedicated" fields of expertise hindering the communication between them. As design projects are becoming more complex, the need to reorganise the design process and the communication between the involved disciplines is becoming crucial. This communication is described by the exchange of data and it often involves the translation of one form of model into another. One of the most difficult tasks that modern engineering practices are summoned to confront is the transfer of geometry from the architect's model to one that can be used for structural assessment. According to Kostler, "Transfer of geometry generated in the architects design tool into structural engineering analysis was problematic and often resulted in simple geometric data without relational or dimensional information" [1].
The term "interoperability", initially defined only for IT systems or services can reasonably now apply to construction technologies. The Institute of Electrical and Electronics Engineers defines interoperability as "*the ability of two or more systems or components to exchange information and to use the information that has been exchanged*". [17]

A well-established platform that enables interoperability between building industry related software is described in the Industry Foundation Classes data model (IFC format). The format is commonly used in Building Information Modelling (BIM) and is used by a variety of software vendors to enable users to exchange data between different software [36].

In the context of this thesis, a case study on an existing building project has been carried out by the author, in order to distinguish the issues arising from the collaboration between the disciplines that are involved in the design and construction of a complex building. The case study describes the process followed from design to fabrication of the grid-structured roof of MediaCite in Liege and focuses on the collaboration between the members of the design team and the data-exchange methods used to realise the project [18].

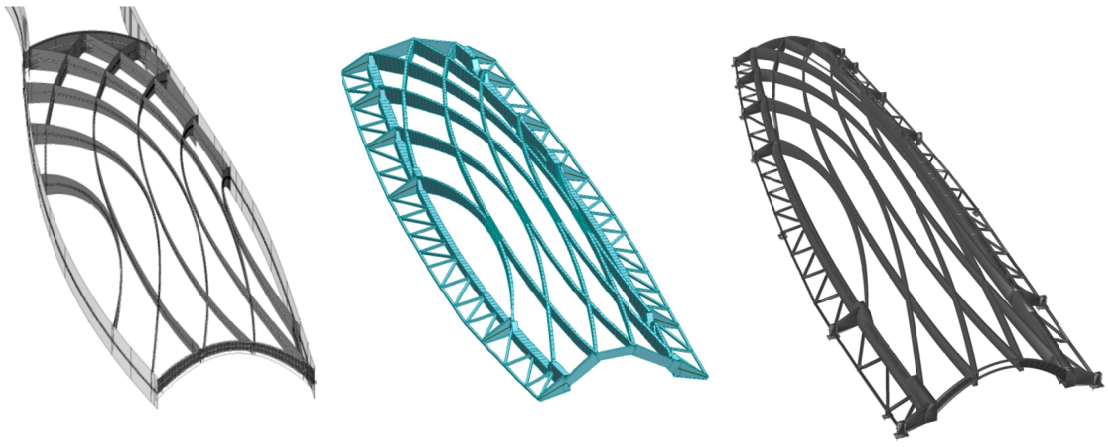


Figure 5 - The three different software platforms used by the design team (Images courtesy of: Ron Arad Architects, Buro Happold Consulting Engineers, and IEMANTS STAALCONSTRUCTIES respectively)

The aforementioned project reflects a major challenge in designing and fabricating modern complex structures and at the same time outlines some common collaboration problems that are encountered in practice. The use of distinct design models and different software platforms by the design team members is something common (see Figure 5). However, when it comes to non-conventional shapes, translating and exchanging information that can be universally understood and implemented is an arduous process that can often limit the possibilities of an optimum result. This study outlines the process of translating each information model for use by each different discipline from concept to fabrication. Additionally, it marks the difficulties that emerge, the data loss and the extra time needed to resolve arising technical queries. What is more, it shows the possibility for the process to be automated in a large extent and suggests the need for a common platform or framework to administer this collaboration. Further description of the above work is not within the scope of this thesis, though it is available for reference.

Among others, this thesis aims to improve this field of communication between multiple disciplines involved in construction, by introducing a series of tools that either connect software to interchange information or translate data to be used in different forms. In addition, it proposes the advance creation of interoperability frameworks to aid this process by setting up common information exchange platforms for efficient communication between the design teams.

3 An Interactive approach to Structural Analysis

In this section, the approach followed to generate tools that aid the interoperability and interactive structural analysis is presented and discussed. Two examples are used to illustrate this process: a simple one that handles the case of a two dimensional truss and a more complex one that applies to the case of free form surfaces. In both examples, a parametric design software is combined with a structural analysis software using computer programming to enable real-time exchange of data and visualisation of analysis results.

3.1 Parametric definition of a 2D Truss

To enable a multi-objective structural design, a small software project was assembled in order to demonstrate the possibilities and extents of a collaboration between generative design and structural analysis tools. The aim of this software was to achieve interoperability between parametric design software and a structural analysis package. For this purpose, McNeal's Grasshopper 3D and Autodesk's Robot Structural Analysis [19], were linked together using the C# programming language. In particular, Grasshopper3D is used to parametrically generate geometry which is sent to Robot for structural analysis, its results being returned to Grasshopper for visualisation or to perform optimisation tasks.

The functionality of Grasshopper 3D (GH) can be extended by writing code in C# or VB DotNet programming language to create custom components. In parallel, Robot Structural Analysis (RSA) allows the interaction with other software and the use of its Calculation Engine through an Application Programming Interface (API) [20].

A distinction should be made between the way that the two applications understand and control geometry: when modelling a structure, a set of notions should be taken into account in order to facilitate the process of defining and analysing a certain structure. These notions can be defined as the structural modelling entities and include nodes, bars and panels and match real building elements as foundations, beams, slabs and so on. Some of these entities also exist in the pure geometric model, sometimes with different naming, and in fact determine a similar notion. For example, two points can describe a line and in the same way, two connected nodes describe a structural bar. What makes a significant difference in the representation of a structural model is the need to attach structural attributes to each of its elements, while the geometric model can be purely described by its topology. In addition, the numbering of each element and its global orientation in relation to its local axis definition are crucial points in the definition of a structural model (Figure 6). A simple bar element for example, is labelled with a number and is defined by its two interconnected nodes ***i*** and ***j,*** each one numbered individually in the global system. The accurate definition of nodes is important in structural analysis since that is where all the calculations occur.

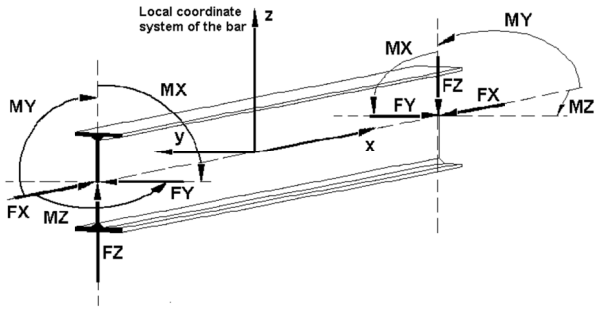


Figure 6- Local Coordinate system and sign Convention, Image courtesy of Robot Structural Analysis v.18 User's Manual

In this context, when drawing any geometry to be assigned structural properties, care must be taken in defining its connectivity and naming. Thus, each node that defines geometry in GH needs to be modified accordingly to be read by RSA. For example in order to translate a line to a structural bar, the line's *start* and *end* point coordinates are extracted from GH and are used to define two new nodes for RSA. Those nodes are then used to define a structural bar since there does not exist an implicit way of translating a geometric line to a bar. Each structural member in RSA is defined or controlled using an appropriate interface, which is included in the API library, called Robot Object Model [21]. An interface is a software structure comprising a set of data, defined as attributes or members, and operations that can be performed which are called functions. These interfaces and their functions match the operations that a user follows to model and analyse a structure in the actual software environment. A node for example can be represented by the *RobotNode* interface, which among others includes three real numbers for x, y and z global coordinates. Each node can be managed using the *RobotNodeServer* interface, which for example includes the *Create* function for creating a new node.

For the definition of the Parametric Truss, its geometry was first defined parametrically so that all of its attributes (at an initial stage only the geometrical ones) could be controlled in an interactive manner. The attributes included the truss' overall span, depth and number of bays. The coordinates of the points comprising its geometric connectivity were stored in ***x***, ***y*** and ***z*** programming lists and were then used in sequence by the *RobotNodeServer* interface for structural nodes to be created. Using the interface, each new structural node is created by inputting its global number value and its ***x***,***y***,***z*** coordinates. Each subsequent node is then connected to form structural bars using the *RobotBarServer* following the same connectivity sequence used in defining the geometric model. Relatively, the inputs needed by the *RobotBarServer* interface are each bar's global number value and its start and end node number values.

The GH geometry is then fully associated with RSA geometry and properties, consequently geometrical parameters imply structural modelling parameters. At this stage, structural constraints such as supports, bar sections, material properties and loads need to be included in the model. This is done by using RSA's relevant interfaces and members in a

similar way, as the user would define them using the software's Graphical User Interface (GUI). When a structural model includes all the information needed to be analysed, the calculation engine interface (*RobotCalcEngine*) can be called and the desired results can be generated and returned inside the GH custom component. The results are returned by utilizing the appropriate interface, depending on the type of the results and the reference to the structural entity. In the current case, nodal deflection and bar force data was returned from the analysis. These results can now be used as parametric outputs to enable structural analysis visualisation in Rhinoceros 3D environment (see Figures 8 and 9).

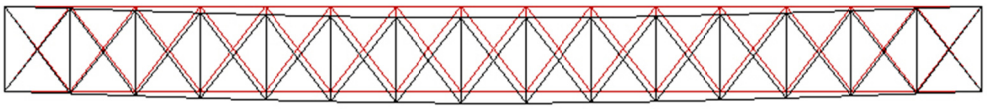


Figure 7- Visualisation of deflection under self-weight in Rhino3D

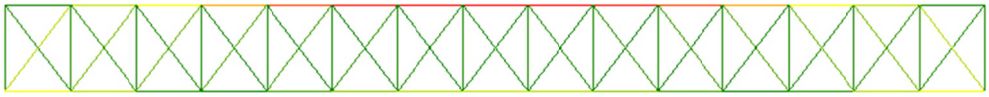


Figure 8- Visualisation of the distribution of forces in each element under self-weight in Rhino3D

This tool's interactivity is further extended by employing GH's parametric capabilities, that is to assign parametric control to structural variables that were used to model the truss. This enables any line of load or any curve drawn in Rhinoceros's environment to be used to apply a distributed load case on the truss' top chord and can then be in addition controlled in real time by curve control points. In a similar manner, the truss' support points can be moved by using a GH numerical slider in respect to the original edge supports (see Figure 9).

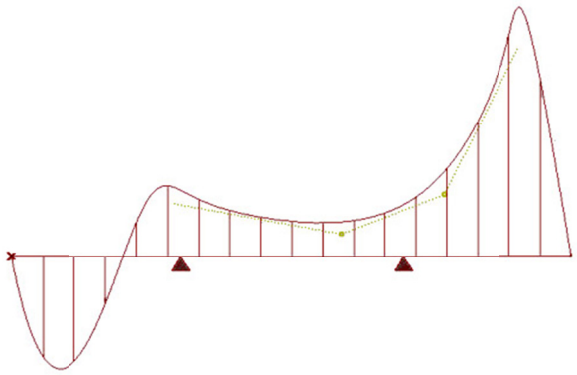


Figure 9 - Any arbitrary load case curve can be drawn in Rhino 3D and the support position can be changed using a numerical slider in Grasshopper 3D

A wide range of parameters is possible to be added and controlled within GH's custom component for the truss and in the current context, such control is given by selecting multiple load cases or load case combinations, different truss types (Brown truss, Pratt truss and Vierendeel truss), section type selection from RSA steel section library and truss length (see Figure 10).

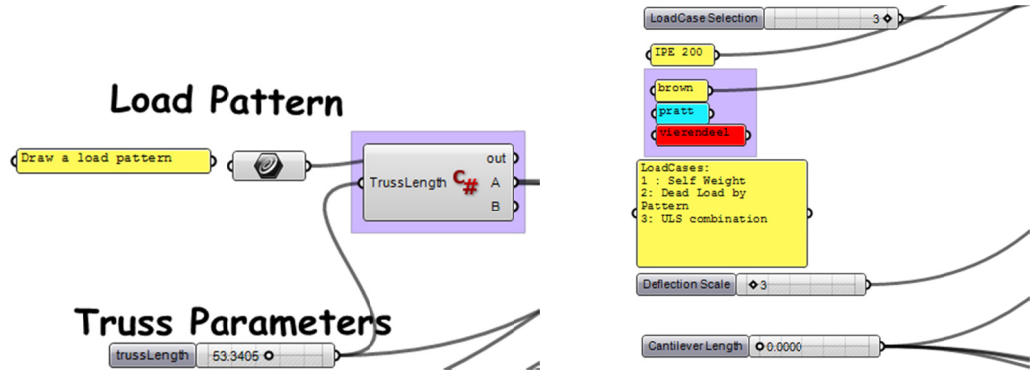


Figure 10 - Defining the truss' parameters in Grasshopper's sketchpad. The variable parameters include the truss length, design cross-section, supports position, type of truss (Brown truss, Pratt truss, Vierendeel truss) and load case

3.2 Interactive Analysis of a free form surface

The previous example illustrated the link between a parametric design and structural analysis software applied on a simple 2D problem. A more complex case is investigated in this section, which relates to a free form surface. It is logical that such surfaces demand a larger effort to be modelled, analysed and assessed and thus being able to control geometry changes with instant result feedback would generate an efficient tool for the designer.

Grasshopper 3D supports NURBS, which is the most suitable method for the digital representation of free form surfaces. NURBS have an inherent *u,v* coordinate system suitable for architectural or structural applications (surface subdivision or panelising for example), while allowing control of complex forms and highly curved shapes by increasing the number of control points without affecting the surfaces' global geometry (see Figure 11).

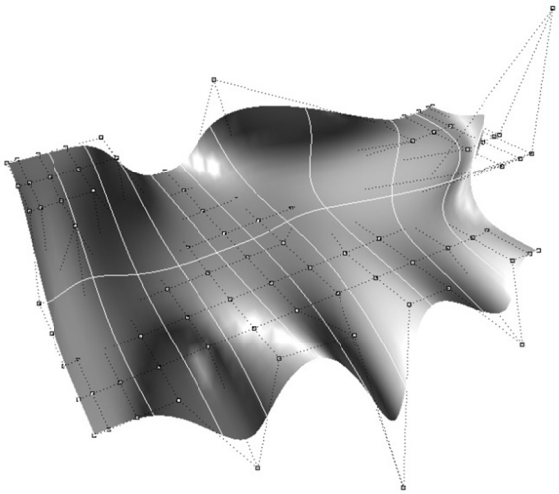


Figure 11 - NURBS surface representation in Rhino3D, showing control points, the resulting continuous surface and lines of constant U and V coordinates

Although the initial modelling and creation of the surface to be analysed is done using NURBS for the reasons discussed above, a triangular mesh is used in the context of the

current research. Meshing is a common method to overcome the intractability of continuous problems by *discretising* them into generic elements whose behaviour is better understood. A Mesh is defined by quadrilateral or triangular Faces (flat facets) bounded by Edges which in turn are defined by Vertices (points) in ***x***,***y***,***z*** coordinates in their two ends. The Finite Element method, which is extensively used throughout this thesis research, is based on this approach. In fact, a Finite Element Analysis (FEA) can be defined as "*a general discretization procedure of continuum problems posed by mathematically defined statements"* [22]. The mesh is an approximation to the continuous surface, touching it only at the vertices. A fine mesh, with smaller but more numerous triangles, is more accurate, but also more expensive to process.

Therefore, as a first step to the process of analysing a free form structural surface, specifically NURBS surface, it needs to be converted into a triangular mesh. A standard GH component is used to mesh any surface that is either drawn in Rhino 3D or described parametrically in GH. The mesh settings are controlled by the component to shape more uniform discretisation of the surface. The mesh *aspect ratio*, which ensures that the ratio of width over height of each quad is less than or equal to the value set [23], is set to 1.0. The faces' maximum edge length is manipulated to achieve either better approximation quality or computation speed. GH's mesh output is a connectivity matrix of vertices, edges and faces. This information needs to be decomposed in elements that can be read by the structural analysis component, which, as was mentioned in the previous example, recognizes data in a different manner. The mesh vertices are translated in a list of points in 3D space, which are then translated in structural nodes inside the analysis component. Up to this point, no relation between the nodes exists only an unreferenced point cloud is drawn in RSA. The connectivity of each face is then used to create arrays of nodes by selecting the three points that comprise each face out of the list. Finite elements can then be created by utilizing the arrays created using the *FiniteElems* interface in RSA. Structural properties need to then be applied, which are treated in a way that allows them to be controlled parametrically by the user through the GH graphical interface. The properties relative to FE and essential to the analytical process are the type of the elements, which are set to planar 3-node elements with constant thickness, the material that either is preset from RSA's library or can manually be defined by its properties and the element's thickness. The model's supports are initialised to the edges by selecting all the nodes that lie on the mesh's outline. The border of the mesh is isolated by utilizing the topological information of the mesh by accessing Rhino.NET SDK [24]. A routine is formed to run through all the edges that form the mesh's topology and select the ones that are "naked", which means that they are only connected to one face. The vertices that belong to those edges are output to the analysis component and are selected as nodes that contain support information. The topological information is lost when the mesh is translated to FE for the structural modelling and therefore the index of each support vertex needs to be found from the list of nodes that are already drawn in the model. Objects needed to be selected individually for attributes to

be assigned on them. That would need recursion loops running through each list of objects, selecting and assigning properties, which is a slow programming process. RSA can select multiple entries by treating them as text values (strings) and assign them attributes. Therefore, in order to improve the calculation performance where possible, numerical variables were converted to strings and were sent for text selection in RSA.

After the structural information database is complete, the calculation interface can be called and results can be output for each finite element. It is important at this stage, that the structural coordinate system be aligned with the environment in which the results are to be visualized. For that reason, the user is able to select the direction that the FE results are aligned to.

For the purpose of graphical representation of the FEA results, each face was translated into a triangulated surface while keeping the topological information of the mesh. Consequently, the analysis results are mapped with each surface facet and their values can be tagged accordingly or coloured using a *GH Gradient* component.
This results to the interactive representation of the impact of change on the geometry or the constraints of the initial NURBS surface in Rhino's Viewport (see Figure 12).

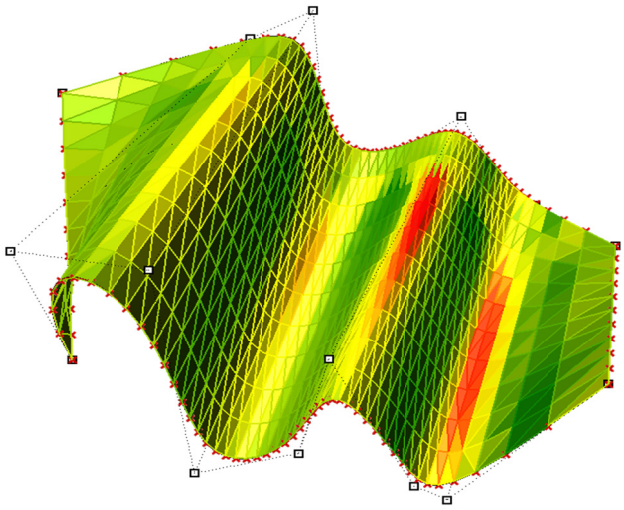


Figure 12 - Graphical representation of surface stress ratios in Grasshopper 3D

In a later stage, the option of adding support points to represent columns, rather than only having supports on the perimeter, was implemented in the tool. Using GH's *surface – curve intersection* component the point of intersection between the column and the NURBS surface was found. By using the mesh's topology interface, the index of the faces containing the supports were isolated and removed from the mesh. Three new faces were then created by including the new vertices, thus subdividing the original mesh. The new mesh is input into the RSA component following the aforementioned procedure for analysis. The

user can switch between the two types of support conditions using a *Boolean* toggle switch in GH (see Figure 13).

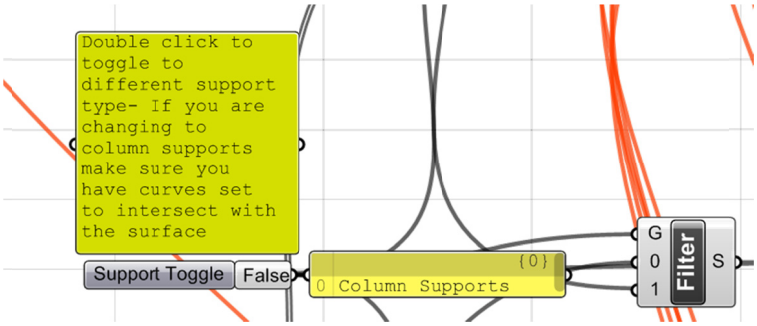


Figure 13 - Support type toggle switch

4 Interactive Form-finding

In a second phase, ways to extend the use of the relation between *generative* design and structural analysis (established in Chapter 3) are explored, in order to investigate efficient structural forms. An attempt is made therefore to apply optimisation techniques to the custom components in order to determine the best possible outcome to the given problem while satisfying certain constraints. Through these techniques, *parametric form-finding* is introduced, as a means to explore multiple improved approaches, rather than seeking for a single automated solution.

4.1 Parametric Truss Form-finding

In the case of the parametric 2D truss, a framework is defined for the designer to explore multiple definitions to the problem while being able to instantly visualise the results. At this stage, the resulting output for each different definition is used to find a better form of solution depending on the constraints that are set.

A well-established method leading to a more efficient form of a truss is to adjust its shape to its stress conditions. Trusses are used to bridge spans and therefore the most vital form of force that they are subject to is bending moment. Bending moment introduces axial forces on to the chords of the truss and consequently members are more stressed in areas of high bending moments.

In mechanics, the stress induced in a member of constant section under pure bending force equals to:

$$\sigma_{xx} = \frac{M_{yy}}{I_{yy}} z$$

where σ_{xx} is the normal stress, M_{yy} is the applied moment, I_{yy} is the section's moment of inertia relative to the bending axis and z is the distance from the members neutral axis to the point that the stress is calculated(usually this is taken as the cross sections mid-depth as it is where the maximum stress occurs). The above relationship proves that the distribution of stresses in a member under pure bending is determined by the bending moment that it is subject to and the section's moment of inertia. Therefore, a cross section is able to withstand larger moment force with an increase of its moment of inertia. A sections moment of inertia describes the distribution of its "mass" relative to a coordinate system and it is defined by:

$$I_{yy} = \int z^2 dA$$

where dA is a finite segment of the sections area and z is its distance from the stated axis.

It is anticipated therefore that a section is more effective in bending as it is deeper and more area is distributed away from the sections mid-depth axis. That is the main reason for the I –beam being used widely in the construction industry [25].

The principles described above can also similarly apply to trusses, which in turn can be approximated as solid sections by also allowing for a reduction factor for shear and axial distortion since the connection between the truss's chords is not as adequate as the one of a solid beam. The algorithm applied to the GH component adjusts the truss's depth according the initial stress conditions, caused by the influence of bending moment, until the sections of the truss chords are utilized to the larger extent. This leads to a form of truss associated with the shape of the bending moment diagram (see Figure 14).

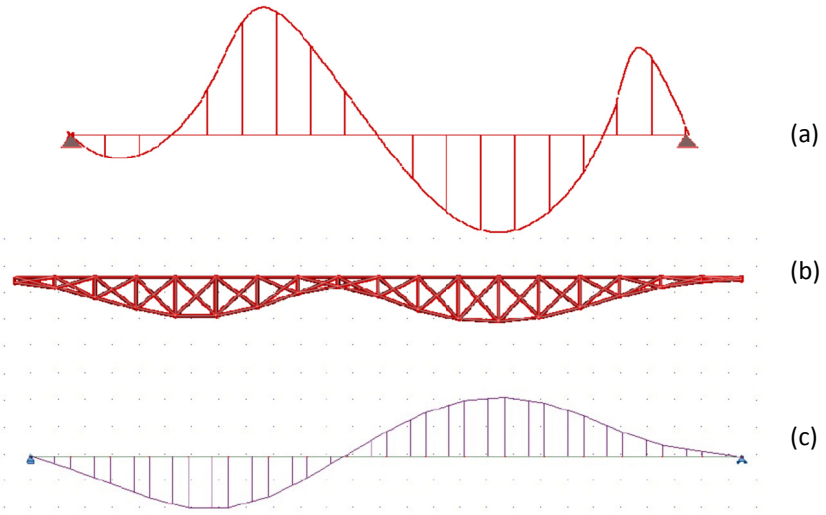


Figure 14 - Resulting output of the algorithm (b) in relation to the bending moment diagram (c) under an arbitrary load case (a)

The designer can alter all the parameters that were defined previously to form the truss and the component will shape the truss according to those parameters. This offers additional control to the engineer while moving away from the distinct optimum solutions to the problem (see Figure 15).

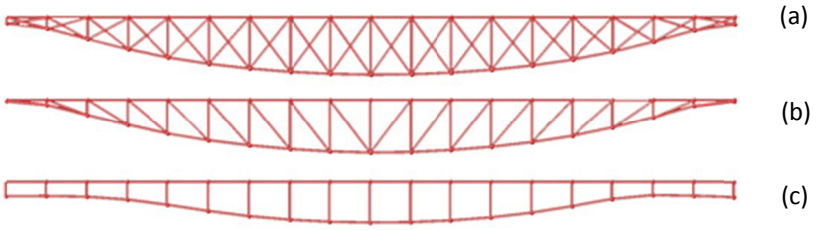


Figure 15 - Resulting optimised shapes for Brown truss (a), Pratt truss (b) and Vierendeel Truss (c) (consisting of fixed joined bars)

4.2 Developing structurally efficient grid structures

The necessity of designing a grid structure in today's modern projects evolves through the need to realise the construction of a free form architectural shape. The usual process followed has as a starting point a fixed free form surface and involves laying an aesthetically pleasing grid onto it, which is then rigorously engineered to utilize the section sizes of its members as efficiently as possible. Detailed information on this approach is described in the MediaCite Liege case study [18] while it has been used in a variety of completed projects, for example Foster & Partners Smithsonian Courtyard enclosure (see Figure 16) [26] or New Trade Fair Milano by Massimiliano Fuksas [27].

Figure 16 - The Smithsonian courtyard grid structure, (Photo: Timothy Hursley)

The above process raises some issues on current grids' structural efficiency and suggests the need for such structures to be efficiently engineered from the start, rather than engineered to meet efficiency following their proposed architectural layout. This thesis introduces a technique for designing grids based on the directions of principal stresses of the continuum shell (that represents a free form surface). This technique is not new in particular and has been previously introduced by Pier Luigi Nervi [12] in the design of ribbed slabs. The idea of aligning a grid structure along the principal stress has been further developed by Winslow [28] and Michalatos [29] for the design of free form shapes. The methods introduced by the latter two are employing surface reparametrisation techniques that were successfully used in a similar manner in computer graphics [30].

4.3 Definition of the problem

Principal stresses are the components of the stress tensor that occur at each point of a continuum, which are purely axial, consequently their shear component equals to zero. The directions at which these stresses occur are called Principal Stress Vectors. These components share the maximum and minimum stress values and ideally, if a grid is aligned along their directions it can replace the continuum.

The influence of the stresses along those directions is evident in the case of a simply supported beam. Figure 17 shows the tension stress trajectories and the cracks that develop in the concrete beam (perpendicular to the former) caused by the material's inability to withstand tension.

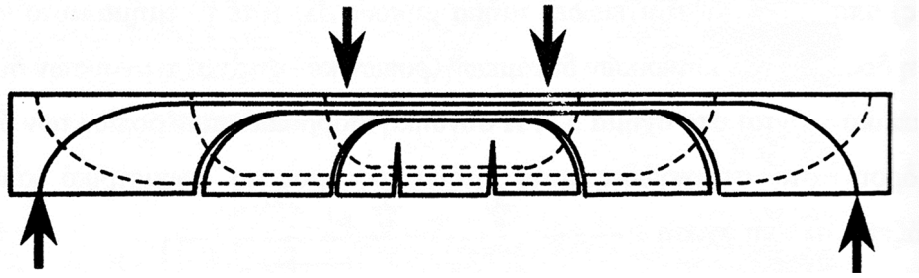


Figure 17 - Cracks developed in a simply supported concrete beam (image: Kotsovos, M.D. [31])

The most efficient way to balance this weakness would be to incorporate steel rods along the tension trajectories developed by the external loading (see Figure 18). However, because such arrangement is difficult to construct the usual reinforcement of a concrete beam consists of straight bars [31].

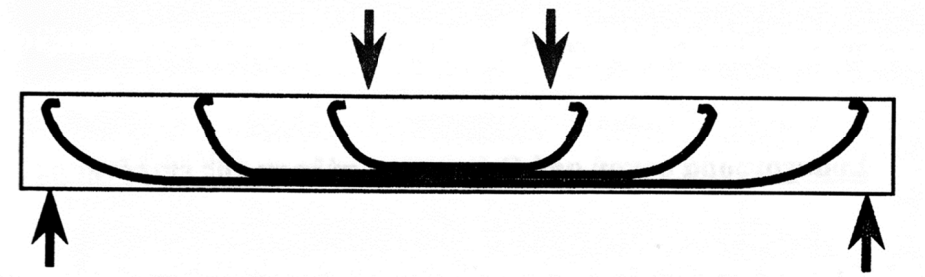


Figure 18 - Theoretical reinforcement arrangement following the tension principal directions (image: Kotsovos, M.D. [31])

In the current context, plane stress conditions will be considered, i.e. cases where one dimension of the continuum is much smaller than the other two are only explored and thus the stresses that act in the through-thickness direction are neglected.

The principal stresses are derived from the global stresses calculated for each element according to the following formulas:

$$\sigma 1 = \frac{\sigma_{xx} + \sigma_{yy}}{2} + \sqrt{\frac{\left(\sigma_{xx} - \sigma_{yy}\right)^2}{4} + {\sigma_{xy}}^2}$$

$$\sigma 2 = \frac{\sigma_{xx} + \sigma_{yy}}{2} - \sqrt{\frac{\left(\sigma_{xx} - \sigma_{yy}\right)^2}{4} + {\sigma_{xy}}^2}$$

$$\tau_{max} = \left|\frac{\sigma_1 - \sigma_1}{2}\right|$$

$$\theta = \frac{1}{2}\tan^{-1}\frac{2\sigma_{xy}}{\sigma_{xx} - \sigma_{yy}}$$

where σ_{xx}, σ_{yy} are the stresses calculated in the X and Y direction respectively and σ_{xy} is the shear stress occurring on the plane with normal aligned with the x direction, acting on the Y direction, and σ_1 and σ_2 denote the two principal stresses, τ_{max} is the maximum shear stress, θ is the angle between the X –X axis and the direction of σ_1.
The formula that defines the angle θ also proves that the two principal stresses σ_1 and σ_2 are perpendicular to each other.

4.4 Approach to the problem

The initial idea for solving the problem was to use the framework of the link between GH and RSA to extract analysis data, specifically ϑ angles, and then use a step-wise method to plot lines that follow the directions of principal stresses.
However, to a better understanding of the plotting procedure, a similar notion was used to approach the problem. It was decided for a starting point that principal curvatures (PC) directions were used instead, in order to prove the viability of the step-wise method. Principal curvature directions comprise orthogonal linear vector fields and plotting lines along them would be a similar task to plotting lines along stress directions (even though this does not make sense in the composition of a structurally efficient grid). The reason for that is the inherent capability of GH to calculate principal curvatures along any surface by using its SDK interface and employing them would be a much faster way to understand the issues arising. Principal curvature is the maximum and minimum normal curvature directions at any point along a surface. They are perpendicular to each other and independent from the surfaces' u and v directions (see Figure 19).

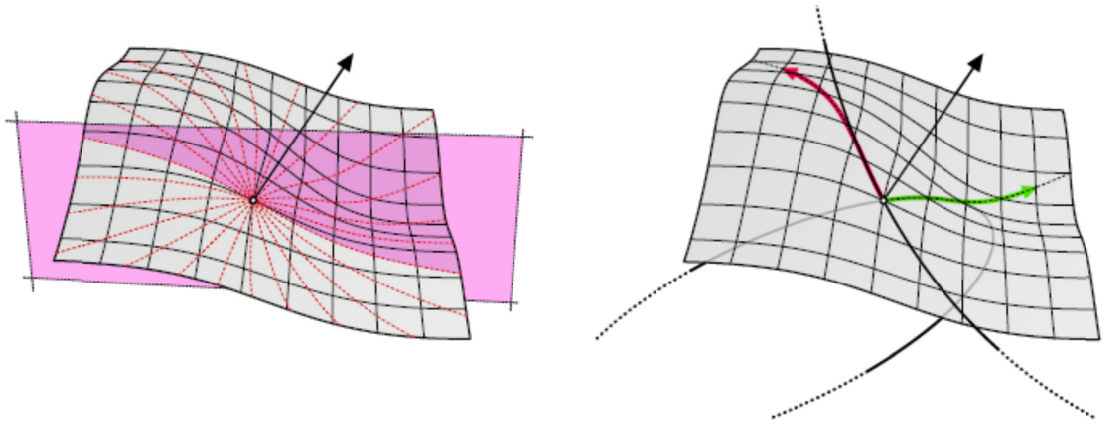


Figure 19 (a) The set of all normal curvatures (b) Principal curvatures at a point on the surface

(Images: Rutten, D., RhinoScript 101)

Principal curvature directions can be calculated by a built-in component in GH given a u-v coordinate on the surface. By subdividing a test surface by u-v parameters, a point grid is

created at which PC directions are determined. A similar procedure is coded inside the C# custom component.

The process of plotting PC trajectories along the surface involved the following steps:

(1) start from evenly spaced points along the surface's border
(2) calculate the principal curvature directions at that point
(3) following the direction for a small step (length) to draw a line
(4) finding the line's end point and projecting it on the surface
(5) Calculate the u-v coordinates of the projected point
(6) Continue the loop until reaching the end of the u and v domain.
(7) Polylines are then plotted from the previous list of points
(8) The minimum principal curvature is then found by rotating the former by 90°.

The plotting is repeated for every u and v coordinate along the edge of the surface.

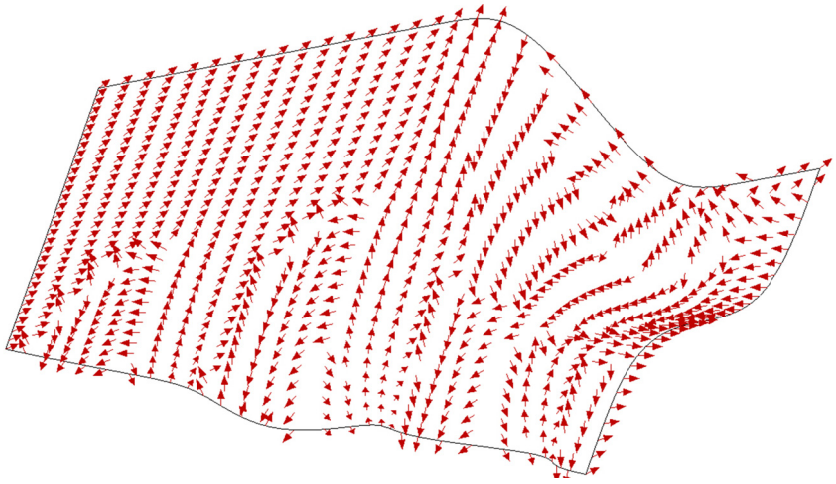


Figure 20 - Maximum principal curvature directions on an arbitrary surface

By observing the directions output, one can easily notice that they make sudden turns of 90° or 180°, creating a complex and confused field (see Figure 20). A possible solution for that would be to smooth the vector fields, as it is usually the case for texturing or patterning surfaces, but that would cause the directions to lose their sense in the case of principal stresses. Instead of smoothing, an algorithm was created that could choose the best direction among the set of four (also adding the two opposite principal directions). The algorithm selected the direction that formed the least angle with the direction selected on a previous step. By that, the trajectories followed the smoothest possible route while they continued to lie along the principal directions field (see Figure 21).

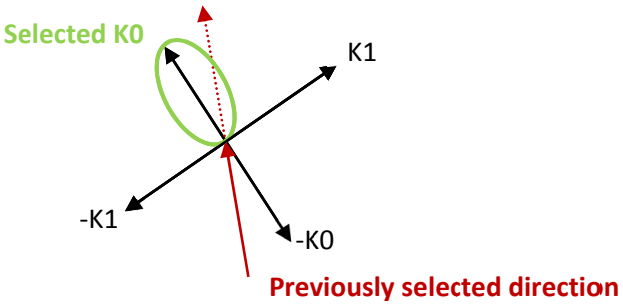


Figure 21 - Vector selection algorithm

where K0 and K1 are the minimum and maximum principal curvature directions respectively.

The angle between the four possible directions and the previously selected direction is found by calculating the Dot Product of each pair of vectors. This method also overcomes the problem of determining the right direction at umbilical areas [32] (areas where equal stresses) occur in both directions. The trajectory plotting depends greatly on the starting vector, i.e. the first principal direction calculated on a single point on the surfaces edge. The first direction does not necessarily point inside the surface and in that case, a trajectory is not plotted. To correct that, the starting vector needs to be oriented so that it points inside the surface, or in cases where that is not possible, an arbitrary starting direction is used (the vectors would still follow the correct direction in the next step).
Figure 22 illustrates the principal curvature trajectories plotting along the test surface for an edge division of 0.5 linear units. The starting vectors are tangent to both directions (K0 or K1) at every starting point.

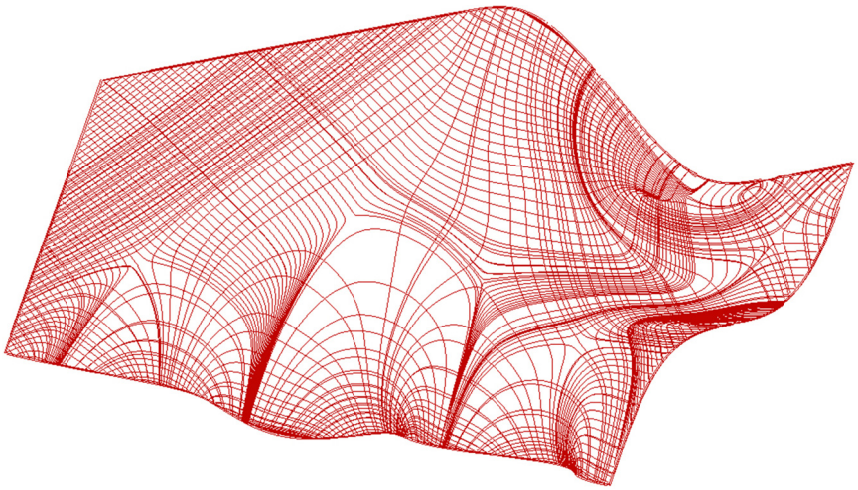


Figure 22 - Principal curvature trajectories plotted on the test surface

4.5 Principal Stress Trajectories

The previous example illustrated the possibility for a grid to be created by following directions tangent to a vector field by using simple step-wise methods. A similar but more sophisticated method is applied at this stage by employing Principal Stress vector fields. Contrasting the principal curvatures case, principal stress directions are calculated using a Finite Element analysis and thus are not resulting from direct functions in Grasshopper but are imported from Robot Structural Analysis. One of the large differences in approaching this case is that principal stress directions are calculated on discrete planar segments, rather than on the NURBS surface itself, as it was the case with principal curvatures. FEA gives one stress result per triangle; there are therefore sharp changes in direction at the triangle edges which did not exist in the case of principal curvatures. In addition, results are reported in element local axes and have to be laboriously transformed back into a global coordinate system.

4.6 Obtaining the principal stress data

The first part of this procedure involved the use of the framework previously established to output values to be used in the plotting algorithm. The same NURBS surface that was previously used at section 4.4 was used for this purpose. In meshing the surface to be analysed, the same settings as before were applied in order to maintain the discretisation consistency. An isotropic material, in particular concrete, was selected out of RSA's built-in material library and was applied on the elements initially with a thickness of 30 centimetres. A linear elastic analysis was then carried out for stress values to be determined. The data was then extracted for multiple layers through the shell's thickness so that bending and membrane stresses were both taken into consideration (see Figure 24). RSA is able to output principal stress magnitudes for σ1, σ2 and σ12 as previously defined and an angle θ. The angle θ is the angle between the maximum principal stress direction and the element's local x-axis. The program arbitrarily selects the local element axes (whose coordinate system cannot be extracted) and therefore needed to be re-oriented by a given direction (see Figure 23). The software allows the local direction to be defined by selecting a global guiding direction. The selected direction was then projected onto each Finite Element and the projected direction comprised the new main local axis (x-axis) of the element.

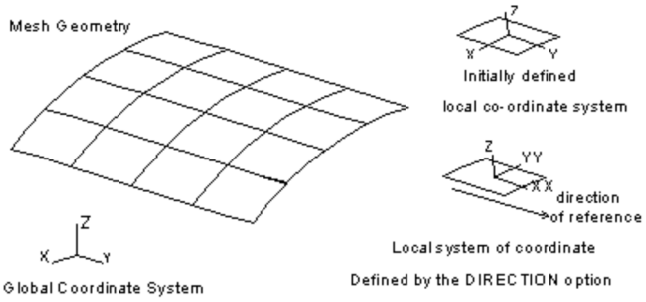


Figure 23 - Finite element local system definition- Image courtesy of Robot Millennium v.18 User's Manual

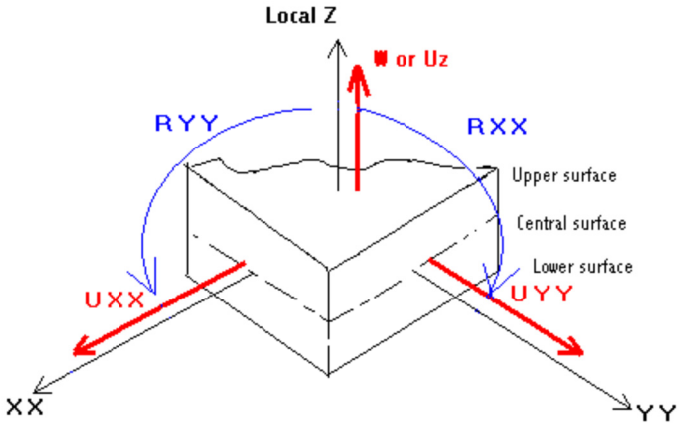


Figure 24 - Finite element sign convention and calculation layers- Image courtesy of Robot Millennium v.18 User's Manual

The principal stress data deriving from the analysis was then sent to GH. In order for the principal stress output to be utilized for the plotting algorithm it needed to be aligned on initial the mesh topology inside Grasshopper. This was done by the following procedure:

(1) Each mesh face was translated to a GH surface
(2) The normal of each surface was extracted
(3) A global direction vector was created (parallel to the direction selected for the results output)
(4) The global direction vector was rotated using each surfaces' Normal Vector as the rotation axis and the *S1* as the angle of rotation.
(5) The vector was then used as a direction to draw a line from each surfaces centroid extending to a finite length.
(6) Each line was then projected back on its surface by using as projection direction the normal vector of the surface.
(7) The tangent of each resulting projected line consists the direction of the maximum Principal Stress
(8) The minimum Principal Stress is then found by rotating the former by 90°.

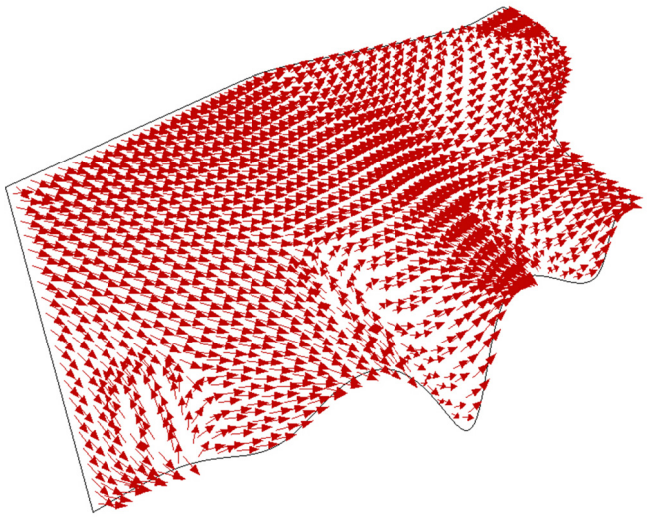


Figure 25 - Maximum Principal Stress Vectors output in Grasshopper

4.7 Plotting trajectories along Principal Stress directions

The principal stress data was initialized at that point to be provided as input to the plotting algorithm. The notion of having to deal with bounded elements (mesh faces that are bound by their edges) initiated the idea of utilising only the boundaries for finding intersections with the principal vectors, i.e. the faces' edges. The idea involved, starting from divisions along the surfaces edge, the following steps:

(1) Drawing short lines from each starting point by following the principal stress direction.
(2) Extending each line to the nearest edge belonging to the same face, (the new line generated is named "*Extended Curve*").
(3) Finding the intersection between the extended line and the faces edge.
(4) Setting the intersection point as a new starting point.
(5) Selecting the right direction by making use of the algorithm developed on the case of principal curvatures.
(6) Continuing the loop until reaching another surface edge.

For the above procedure to be realized, all the principal stress directions had to be mapped to the faces that belonged to the GH mesh. That way, each face had its dedicated principal stress direction aligned on its plane. In addition, all the topological edges had to be translated to lines so that operations, such as extension and intersection, could be applied to them. In order to identify the starting points the "naked faces" (the faces that lie on the perimeter) needed to be distinguished from the global list of faces. That is achieved by searching through the global face list and checking for the edges that only had one face connected to them. The two faces connected to the former would comprise the starting faces and thus the starting directions.

Because Rhino's *extending routine* used infinite segments, it was unable to control the direction at which each line was extended to. Therefore, the starting edge was excluded from the possible extension targets. That was done by creating an Array (of size three) for each face, which contained the three edges (lines) that it was bounded by. Each item of the array was checked whether it was lying on the surfaces edge and if the condition was fulfilled it was excluded from the Array. By making that exclusion every line created from the starting point would only extend to one of the possible remaining edges, while making sure it would not extend back on the naked edge.
At the next step, an intersection interface from Rhino.SDK was used to find an intersection between each possible target item belonging to the array and the *Extended Curve.* The intersection interface returns *Boolean* values of true or false indicating whether the intersection was successful or not. The Boolean value was used to determine the index of the edge at which the intersection occurred and consequently the index of the faces connected to it. By eliminating one of the two faces that was used previously, the next in

turn face (*nextface)* was determined. A new *Array* was then created including all the edges of the next face and the procedure was repeated using the two principal directions that were mapped with the index of that face. At each intersection, the correct direction was chosen out of the four possible directions by assessing its relation to the previous direction selected. The plotting continued until the extended line reached a naked edge and was then repeated for every different starting point, on every different *naked face*. All the intersection points created for a different starting point, were stored to lists and were then connected to form Rhino *polylines.*

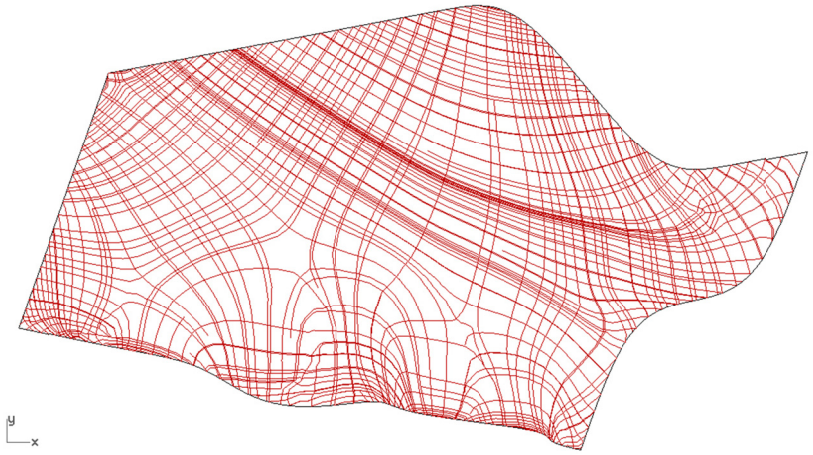


Figure 26 - Principal Stress Plotted Trajectories on test surface

Figure 26 above illustrates the first output of the algorithm tested on the NURBS surface for the initial mesh edge size set 2.0 units. The starting point for each trajectory was the mid-point of each edge at the surface's perimeter. The first approach yielded some serious problems and omissions in the algorithm that needed to be addressed.

The first issue was that Rhino's intersection routine is sensitive to numerical tolerances for intersection and overlaps. As illustrated in Figure 27, tolerance issues occur near the face corner where the intersection fails or occurs in a wrong position. For that reason, it was decided to write a manual intersection routine rather than using Rhinos interface.

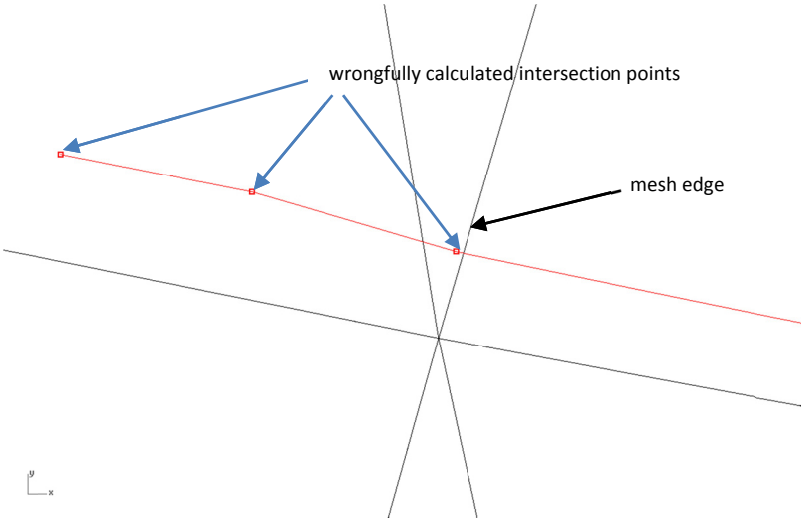


Figure 27 - Problems caused by intersection tolerance

In three-dimensional algebra [32], the intersection of two lines comprising of the points $\mathbf{x_1}=(x_1, y_1, z_1)$, $\mathbf{x_2}=(x_2, y_2, z_2)$ and $\mathbf{x_3}=(x_3, y_3, z_3)$, $\mathbf{x_4}=(x_4, y_4, z_4)$ is found by solving the simultaneous equations:

$$\mathbf{x}=\mathbf{x_1}+(\mathbf{x_2}-\mathbf{x_1})s$$
$$\mathbf{x}=\mathbf{x_3}+(\mathbf{x_4}-\mathbf{x_3})t$$

For an intersection to be found, the condition that the lines are not skew needs to also be fulfilled:

$$(\mathbf{x_3}-\mathbf{x_1}).[(\mathbf{x_2}-\mathbf{x_1})\times(\mathbf{x_4}-\mathbf{x_3})]=0$$

The equations are solved by eliminating s and t with:

$$s = \frac{(\boldsymbol{c}\times\boldsymbol{b})\cdot(\boldsymbol{a}\times\boldsymbol{b})}{|\boldsymbol{a}\times\boldsymbol{b}|^2}$$

Where a, b and c are Vectors in three-dimensional space defined by:

$$\boldsymbol{a} \equiv \boldsymbol{x_2} - \boldsymbol{x_1}$$
$$\boldsymbol{b} \equiv \boldsymbol{x_4} - \boldsymbol{x_3}$$
$$\boldsymbol{c} \equiv \boldsymbol{x_3} - \boldsymbol{x_1}$$

The point of intersection between the two lines is then found by substituting s back into the equation:

$$\boldsymbol{x} = \boldsymbol{x_1} + \boldsymbol{a}s$$

The intersection algorithm was tested to find a possible intersection between the new finite line segment created (by following the selected principal stress direction), now converted to infinite, and the three contents of the edge *Array.* The above equations yield intersection points for infinite lines and therefore the parameter domain in which the intersection is found needed to be checked in order to be ensured that the point was lying within the edges' length. Consequently, where $\boldsymbol{t}$ is the line parameter on every edge:

$$0 < \boldsymbol{t} < 1$$

In addition, the line would not intersect backwards to the face it had started from. The condition was initially maintained by calculating the dot product between the new line created and the one previously plotted ensuring that the two were pointing in the same direction. In a latter phase, this condition was algorithmically refined by just checking that the *edge* onto which the intersection was found was different from the *starting* edge.

Intersecting infinite lines instead of segments had also eliminated the use of Rhino's Extension routine and therefore avoided tolerance issues. For a starting direction to be selected, it had to be ensured that the first principal stress directions pointed inside the surface, as also noted on the principal curvatures' example in section 4.4. That was initially controlled by testing the direction of each principal stress vector with a vector connecting the starting point (on the perimeter edge) and the intersection between the other two edges for every starting face (see Figure 28).

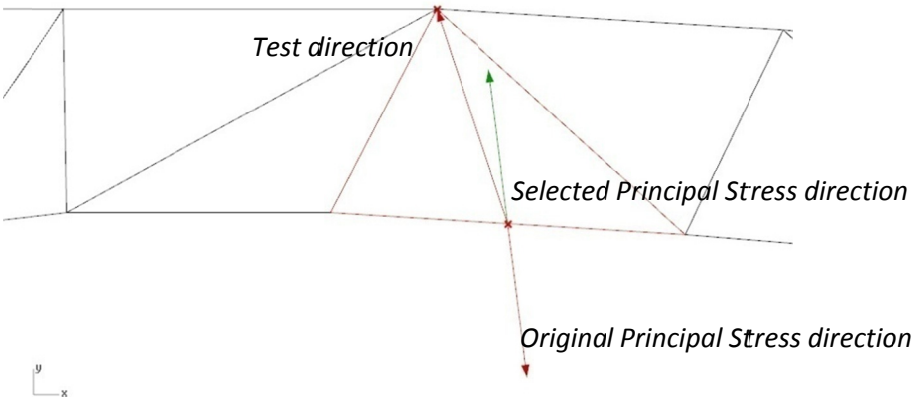


Figure 28 - Initial starting point condition

The above assumptions led to some plotting inconsistencies in the cases where the starting face was of elongated shape as well as at the corner faces where a *test direction* could not be calculated. As a result to that some starting directions where omitted and therefore some trajectories were never plotted.

As a first attempt to tackle the previous problem, attest was performed to determine whether the end point of the starting vector (given a finite length) was inside the face. This was done by comparing the sum of the areas of the triangles composed by the vertices of the tested face and the test point to the area of the face. At a later stage, a simpler method was used, described by the following formula:

$$d = (\boldsymbol{a} \times \boldsymbol{b}) \cdot (\boldsymbol{a} \times \boldsymbol{c})$$

Where a equals the vector tangent to the faces' edge lying on the perimeter b is any vector that certainly points inside the triangle, (the previous Test direction was used in this case) and c is the Principal Stress direction defining the tested starting direction. The number d is positive when the vector c is above a i.e. the resulting normal vectors point in the same direction (see Figure 29).

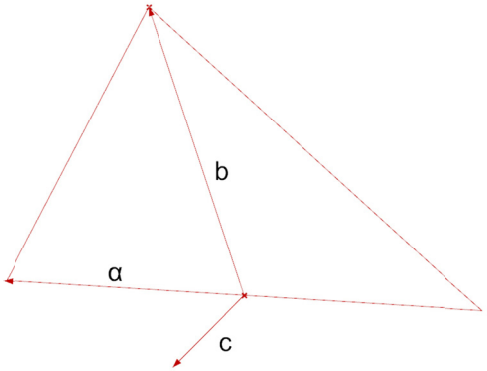


Figure 29 - Condition for reversing the starting direction

The revised algorithm was run on a new iteration and the results are presented on the illustration below (Figure 30):

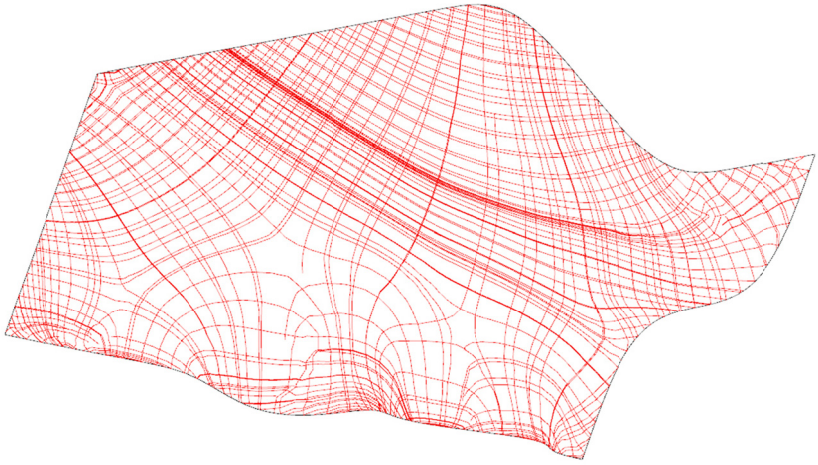


Figure 30 - Principal Stress Plotted Trajectories on test surface

The new algorithm that was created could overcome the tolerance issues created by Rhino's interfaces and was not sensitive to mesh irregularities (in regards to the plotting starting points). While some major issues have been addressed, the new plot has revealed some new errors, which are evident in the previous illustration. One can easily mark trajectories that stop before reaching the surfaces edge. Since the plotting algorithm is face-dependent, i.e. every principal direction is associated with one face and consequently its plane. As a result, to that, at a border of two faces, if a direction other than the one associated with the face which is next in sequence was selected, an intersection could not be found. In those cases, therefore, where the previous plotted line was almost, or nearly parallel to a face edge, it was a possibility for the chosen direction to point back into the face it was coming from.

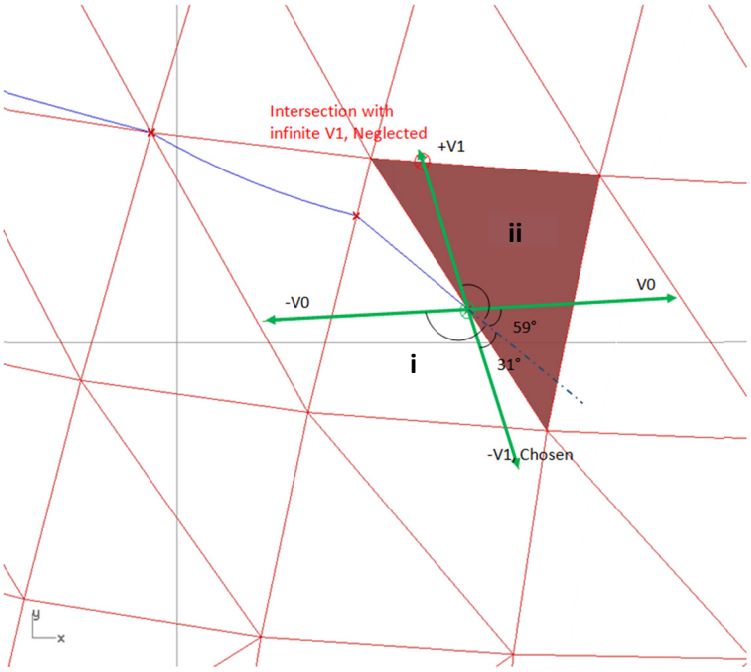


Figure 31 - Direction selection fault

As illustrated in Figure 31, at the point of intersection with a new face, the algorithm is set to choose the best possible direction for the next plotting step. In this example, the direction forming the smallest angle with the previous line plotted (in blue colour) is –V1 (the reversed minimum principal stress direction). The specific direction therefore, points back on the face *i,* which was the face associated with the previously followed direction, without finding any intersection since the Array only includes the edges of face *ii.* As a result to the previous fault, caused by the fact that the intersection routine utilizes infinite lines, -V1 intersects backwards with one of the edges that belong to face *ii* thus streaming a wrong trajectory. By choosing any other direction rather than *–V1* in the current, and in similar cases, would cause further inconsistencies and a plotted trajectory unrelated to principal stress directions. It was therefore decided, to terminate such trajectories at the point where the errors occurred. For that reason an additional condition was added for the termination of the plotting loop, which ensured that the selected direction, defined by the last intersection point, differed by less than 90° from the already plotted part of the trajectory. That caused the sudden termination of some trajectories but on the contrary enabled the successful completion (the contact with the surface edge) of others. In the previous case, where the intersection tolerance issues had not been resolved, a trajectory would continue its path, uncontrolled, until it reached a new edge. For that reason Figure 30 does not show major **plotting** improvements compared to Figure 26.

This issue was unable to be resolved in the context of this thesis, and will be discussed as further work in a later chapter. However, a function was added so that the user can manually override it. As it was discussed previously, the plotting of the trajectories depends greatly on their starting point, which was set to the mid-point of each edge on the surfaces'

perimeter. The division point was now set as a parameter along each edge line, enabling the user to choose a starting point for either all the trajectories or give a different starting point to each one.

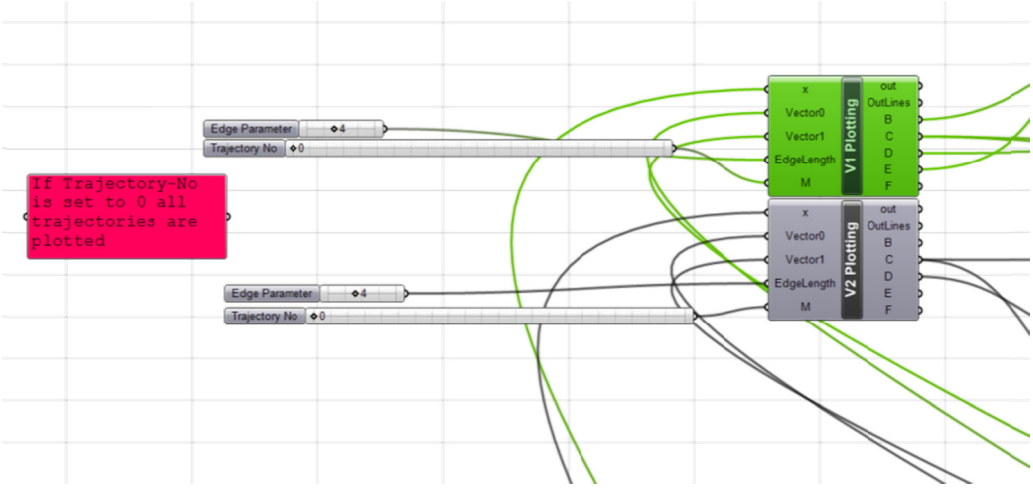


Figure 32 - Trajectories selection sliders

Using the two sliders (see Figure 32), one for each different principal stress direction as a starting direction, the user is able to control the position of the trajectories by manually changing the starting point in order to avoid the errors mentioned earlier or to achieve a more evenly spaced grid. By setting the "*Trajectory No*" slider to zero, all the trajectories are plotted by starting from the parameter of the edge specified. Otherwise, a specific trajectory number can be selected by moving the slider and setting its starting parameter. This procedure can yield more regular grid patterns as shown in the illustrations that follow.

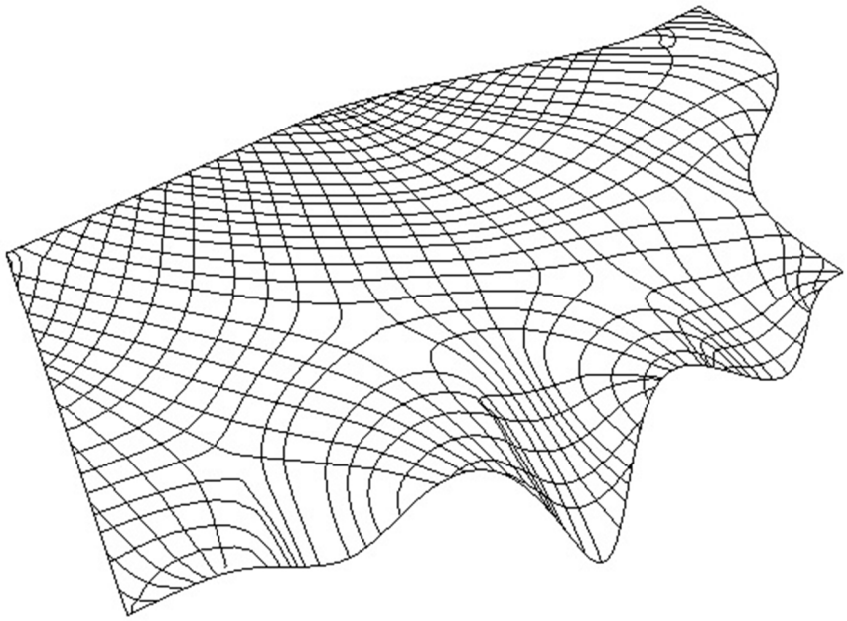


Figure 33 - Evenly spaced grid pattern on the test surface

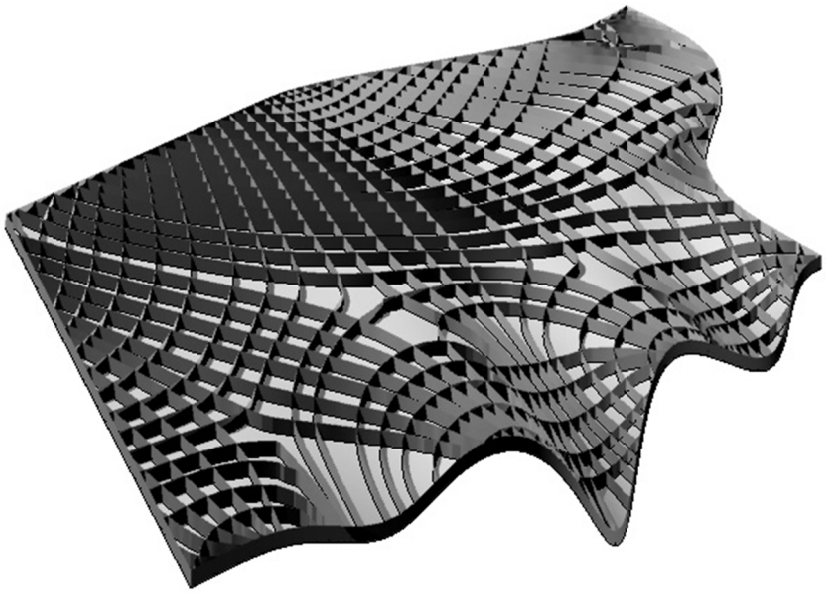


Figure 34 - Rendered visualisation of the extruded grid

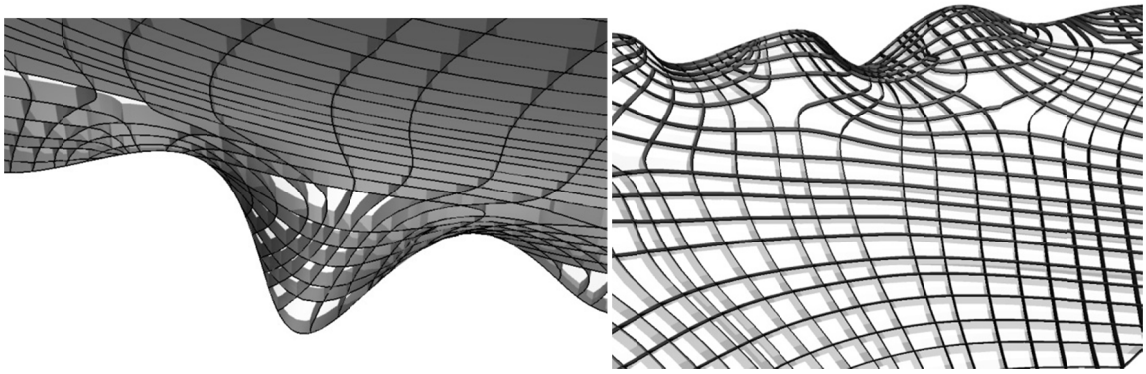


Figure 35 - Rendered visualisation of the extruded grid

The generated grid was then extruded vertically and each trajectory was given a constant thickness in order to form ribs that compose the structure that is illustrated in Figures 34 and 35.

4.8 Random Selection Process

Despite the previous refinements, the algorithm was unable to create an evenly spaced grid without a serious manual intervention by the user and omitted the creation of series of trajectories depending on the shape of the surface. Since the concept of the current application was its universal application to any parametrically defined and controlled surface, it was very important to be refined to meet its initial scope. The major cause of the mentioned faults was that the only control of the position of the trajectories was depended on their initial starting point. Consequently, there exist faces in the meshed surface that have multiple trajectories passing through them and other that have none, thus creating an irregular grid. For the former reason, some of the trajectories never initialize because their starting direction leads them back outside the edge even if they are initially pointing inside

the surface. To investigate this case, the shape of a paraboloid is taken as an example since the stress directions that occur when analysing such a shape under vertical load cases can normally be predicted. Using the current plotting algorithm only the radial trajectories are plotted where one would expect both radial and circumferential principal stress directions. It is evident, from the Figure 36 below, that in such cases (specifically surfaces less planar than the example used previously) the whole set of circumferential trajectories is omitted or just concentrated on the edge (comprising the starting points).

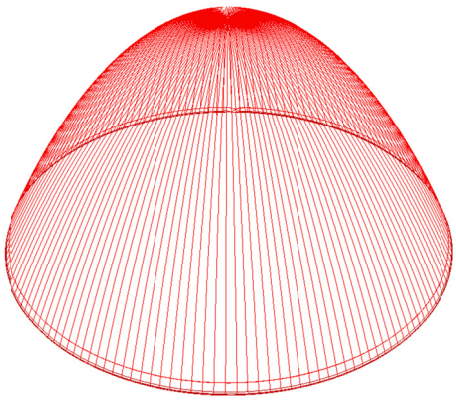


Figure 36 - Plotting trajectories on a Paraboloid – Omitting the circumferential directions (Perspective illustration)

For the reasons discussed above, it was decided that a different approach for setting trajectory starting points should be investigated. Thoughts for alternate starting points created by slicing the surfaces with the use of planes were made but were abandoned quickly since they would also lead to the previous problem, omitting a series of directions. The idea of randomly selecting a starting point, or a face, was then initiated. That incorporated seeding trajectories in all four directions and then eliminating faces which had at least one trajectory per direction passing through it. That would possibly create a more evenly distributed grid where all the faces would be utilized while at the same time ensuring that no more than four trajectories would initiate from the same face.

The general algorithmic process was not changed (i.e. intersecting edges of faces) at this stage apart from the way that trajectories were initialized and the point of initialization. The plotting framework was comprised of two components, one for every set of Principal Stress directions. The new process involves a manual selection of a point on the surface, which is then automatically mapped with the face it belonged to. That point comprised the starting point and the face was mapped to the starting direction of the process. A set of lines with opposite directions (relative to the Principal Stress directions) were then starting from the aforementioned face, intersecting face edges until they reached the surfaces perimeter as described in detail in the previous chapter. The centroids of each face were then calculated and were set as a starting point for their corresponding faces. A list containing the indexes of all the faces was created for every separate component. Following the plotting process, every face that had lines in both directions (i.e. intersections in two different faces) passing through it, was eliminated from the list. A next face was then

selected randomly from the indexed list and the process was repeated until all the faces were utilized in both directions.

This process yielded more solid results while achieving a refined and more evenly spaced grid. In addition, it resolved the problem relating to the sudden termination of the trajectories to a large extent since it was enabling trajectories to restart from the same face if one of their edges was not intersected. The problem persists in some special cases, as seen in the area around the tip of the paraboloid (see Figure 37), which will be discussed in the concluding chapter.

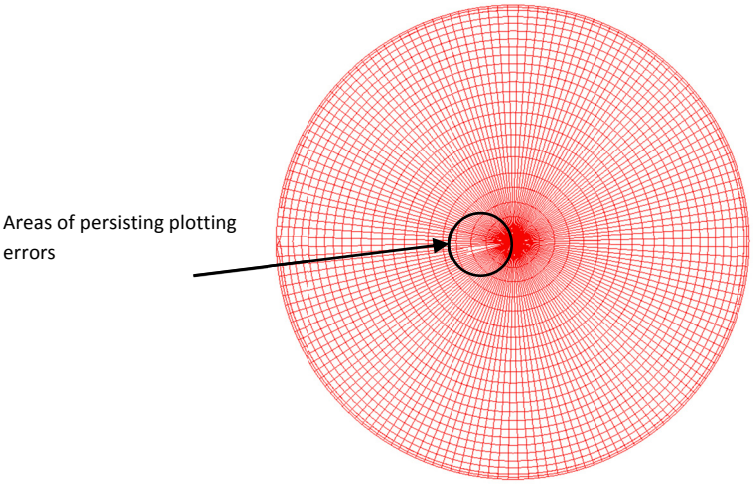


Figure 37 - Plan of the paraboloid surface grid using the revised algorithm

The above illustration shows the case of the Paraboloid tested using the revised algorithm. It is evident that the directions that were omitted in the previous version are now plotted successfully. The next images illustrate the application of the revised algorithm on the test surface using a 2 unit edge length and 1 respectively.

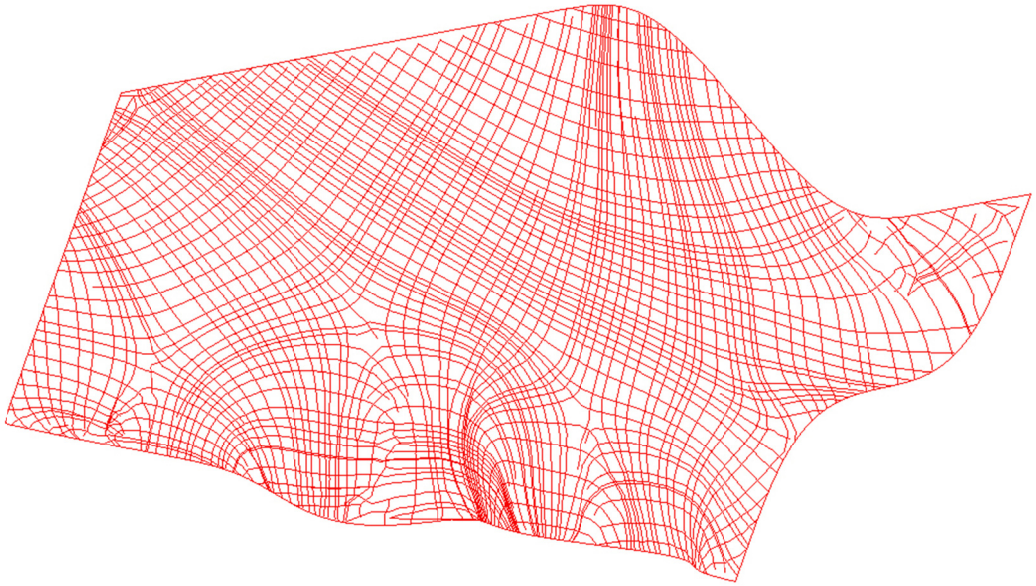


Figure 38 - Principal Stress Plotted Trajectories on test surface

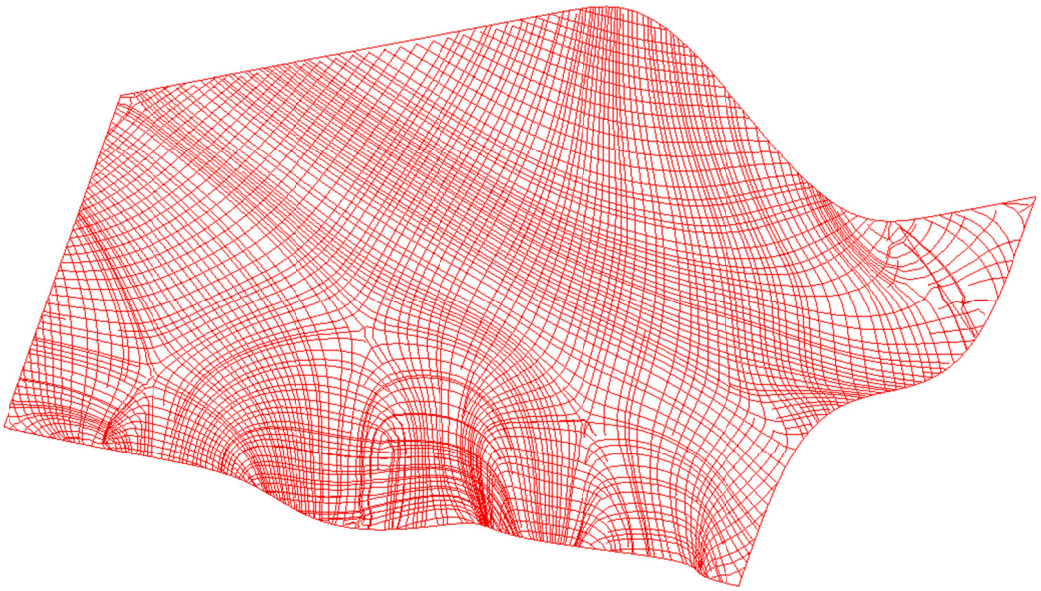


Figure 39 - Principal Stress Plotted Trajectories on test surface (1.0unit edge length)

All the examples investigated up to this point extract solely visualised data and are not thus tested for their structural efficiency. An attempt will be made in the next chapter to create grids to withstand external loading under different conditions.

5 Case Study

5.1 Project Overview

In order to prove the efficiency of the tool developed during the current research, the algorithm was applied on a real design project to create a grid structure. The grid was analysed and its structural performance was compared with other possible grid layouts.

For this purpose, the roof of *Yiapanis Art Gallery* in Cyprus, a project currently under architectural development, was utilised for the creation of a grid structure. The building form was developed through the evolution of a sculptural form and a cooperation between the design architect and a sculptor – client (see Figure 40).

Figure 40 - Yiapanis Art Gallery, Rendering courtesy of Architect – M. Georgiou

The concrete roof is approximately 60 meters long by 10 meters wide and was initially designed to span between a perimeter ring which was supported on edge columns. The roof's surface is doubly curved, and was parametrically defined in Microstation Generative Components (see Figure 41). The surface was re-developed in Rhino 3D so that it could be used with the grid creation tool built in Grasshopper.

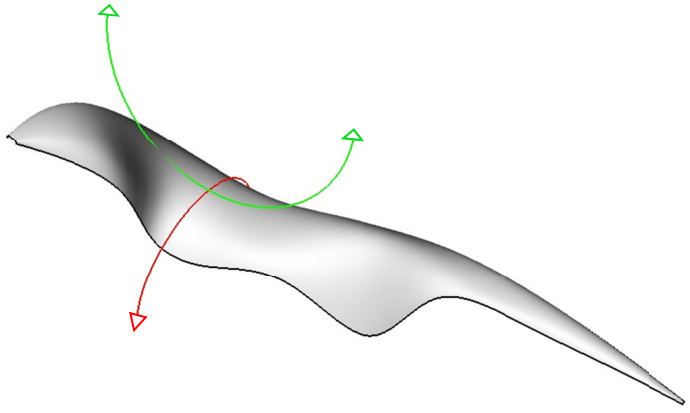


Figure 41 - Doubly curved NURBS surface in Rhino 3D

5.2 Principal Stress grid

The surface was initially set as an input to the meshing component to be translated into discrete geometry for the Finite Element Analysis. The maximum edge length of the mesh was set to 1.0 and the faces maximum aspect ratio was also set to 1.0. These parameters were chosen to produce a fine and evenly-spaced mesh geometry.

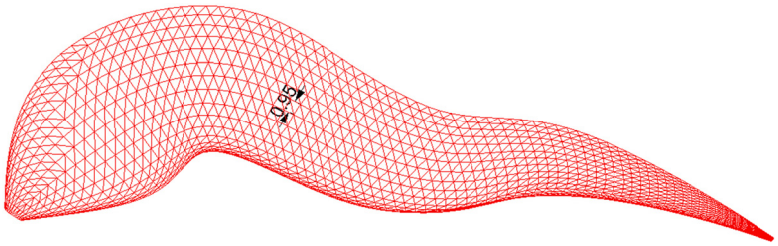


Figure 42 - Roof mesh geometry

The supports were set on each vertex of the surface perimeter and were assumed to be pinned, allowing free rotation in three axes but restricting any displacement. The triangular elements were given a thickness of 300mm and were assigned a concrete material (Young Modulus = 25000 MPa). The load assumed to be acting on the surface was its self-weight (assuming a density of 2453 kg/m^3). The analysis was then run and returned different principal stress results for the three different layers through the Finite Element's thickness (Upper, Middle and Lower).

The principal stress trajectories for the three different cases are plotted automatically by the component, resulting in the illustrations below:

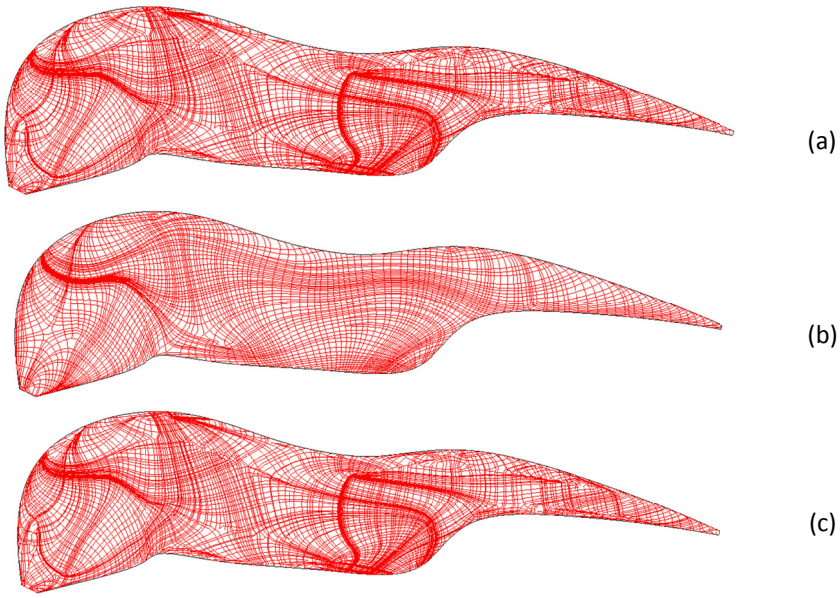


Figure 43 - Principal Stress trajectories for lower (a), middle (b) and upper (c) element thickness layer

From the above illustrations, it is evident that the most appropriate solution, in a visual manner, for a grid application is the one corresponding to the middle-layer principal stress results. This solution partly neglects the influence of bending force, since it is predominantly the outer layers that resist bending moments. Despite this, the impact on the grid's efficiency would be less significant since the roof structure is arch-shaped in its short direction and is therefore mainly subjected to membrane forces rather than bending. The solution corresponding to the middle-layer principal stresses was therefore selected for the next phase since it was easier to post-process and would create a more rational grid.

The plotting algorithm exposed some dense areas where a number of principal stress trajectories converged. Those areas needed to be diluted in order for an evenly spaced grid to be generated. This post-processing was initially applied locally at the areas that needed to be spaced out.

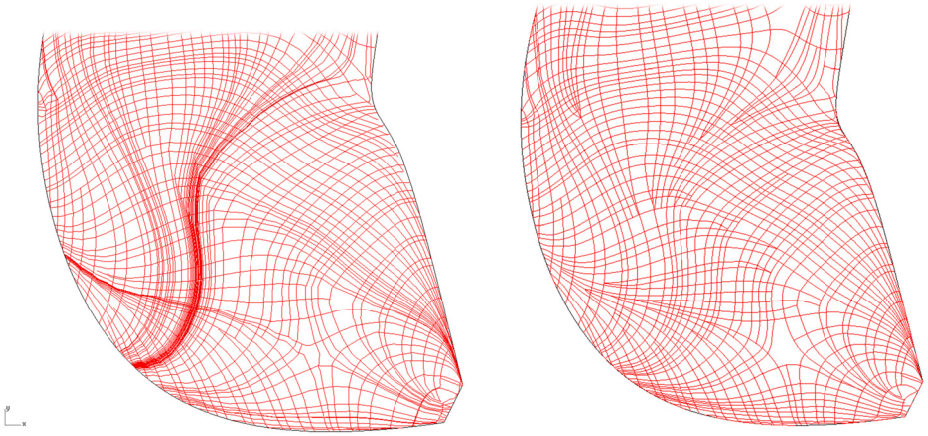


Figure 44 - Areas of trajectory convergence – before and after the spacing process.

This process involved manually setting guide curves (ideally situated at the middle of the dense areas) which were named *Static Curves.* These curves provided a starting point for the spacing procedure. All the remaining curves in that area were *candidates* to either be entirely removed or be merged with the static curves. The candidate curves' indexing was rearranged depending on their distance from the static curves. The static curves were then divided into small segments having their interconnecting points generated. The resulting points were projected on the first in sequence candidate curve. By that, it was ensured that the candidate curves had an equal number of points as with the static curve. The distance between points on the two curves was then checked and each point was either kept in place (if it had an adequate spacing) or projected onto the static curve (if the distance was too small). The candidate curve under process was rebuilt using the series of new points (see Figure 45). At the merging points, the curve was split in two and the merged part was removed. This process generated a "stitch" effect on the trajectories' layout. If the next closest candidate curve had an adequate spacing from the static curve it was selected as the

next static curve. The process was continued until all the curves in the problematic areas were refined or removed (see Figure 44).

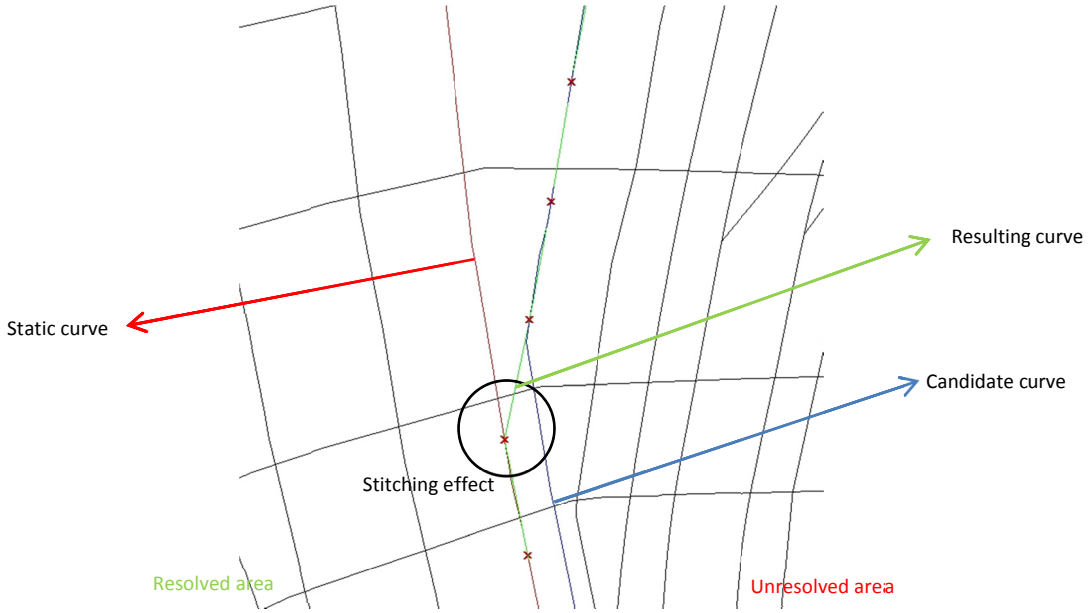


Figure 45 - Curve spacing process

As mentioned in a previous stage, the plotting process took place on the meshed geometry and not on the actual surface and therefore the output needed to be projected on to the latter afterwards. The trajectories were refined at the next step, by rebuilding them using a smallest number of control points to reduce the kinks created during plotting routines (see Figure 46).

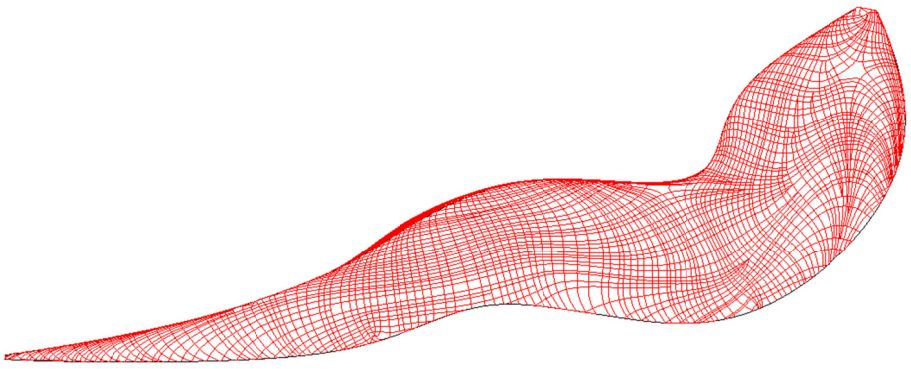


Figure 46 - Refined Principal Stress grid on the case study surface

The process of transforming the refined grid to analysis data for the structural assessment involved the following steps/conditions:

(1) All the trajectories were split at the points of intersection with opposite direction trajectories.

(2) Each segment between two intersection points was *rebuilt*, by setting a curve degree of one and eliminating intermediate points. This produced a number of linear segments which were necessary for the FE analysis.
(3) The *start* and *end* points of each segment were isolated in a GH lists
(4) The global list of grid intersection points was translated into RSA nodes, in a similar procedure to that of the simply supported truss explained in Section 3.1.
(5) Structural bars connecting the nodes belonging to the associated *start* and *end* points list were created.
(6) Contours for the application of loads were created by grouping every four nodes into a closed cell.
(7) Pinned support conditions were assigned to the grid nodes that were adjacent to the perimeter.
(8) A rectangular solid section with dimensions of 100x10mm was created and was assigned to all the linear elements.
(9) A steel material of grade *S275* was assumed for the grid sections.

The analysis model was automatically generated by the GH custom component and was sent to Robot Structural for analysis (see Figure 47).

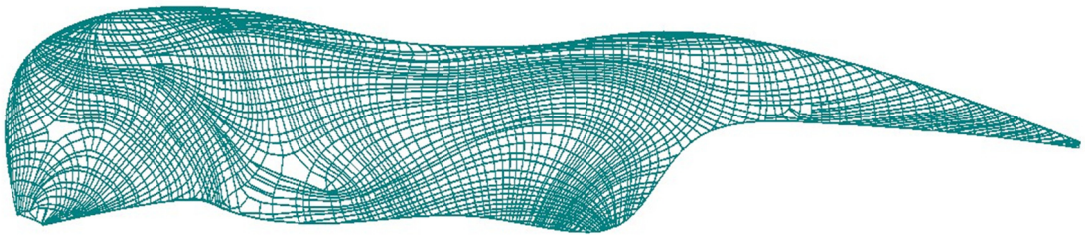


Figure 47 - Structural Analysis model in RSA

Infill panels were incorporated into the model for the application of uniform loads. These panels do not have any impact on the overall stiffness of the model and only work as means of load transfer from the surface to the substructure. A static load of 250kg/m^2 to represent the 100 mm thick concrete surface, that was initially assumed as a finish, 15kg/m^2 services load and 100kg/m^2 imposed load were applied to the cladding panels. The self-weight of the steel grid was added to the structural bars.

The analytical model was calculated using a static linear analysis to determine the vertical deflections and a modal analysis for the assessment of its natural frequency. The structures' natural frequency defines a measure of stiffness and was used to compare the structural efficiency between different grid layouts generated on the case study roof.

The applied loads were combined to fulfil a serviceability limit state condition for the deflection tests and yielded a maximum deflection of approximately *14mm,* which equals a

span over deflection ratio of *1/714* by taking into account that the roof spans effectively along the wide side, which is approximately 10 meters long (see Figure 48).

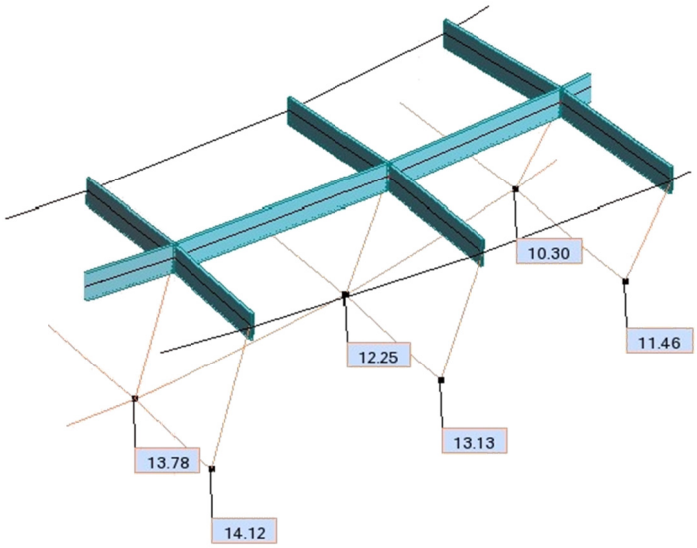


Figure 48 - Maximum deflection of the grid under an SLS load combination case

A modal analysis was then conducted taking in account the first 10 vibration modes and by assuming only the action of the structures' self-weight (see Figure 49). The analysis returned the following results for the first three fundamental mode shapes, which are shown in the table below together with the maximum deflections under self-weight and SLS load cases:

Model Type	Mode No.	Frequency(Hz)	Time Period(sec)	Total Mass(kg)	Self-weight deflection(mm)	SLS deflection (mm)
Principal Stress	1	4.39	0.23	26162.59	5.27	14.14
	2	4.71	0.21			
	3	5.38	0.19			

Table 1 – Analysis results for Principal Stress grids

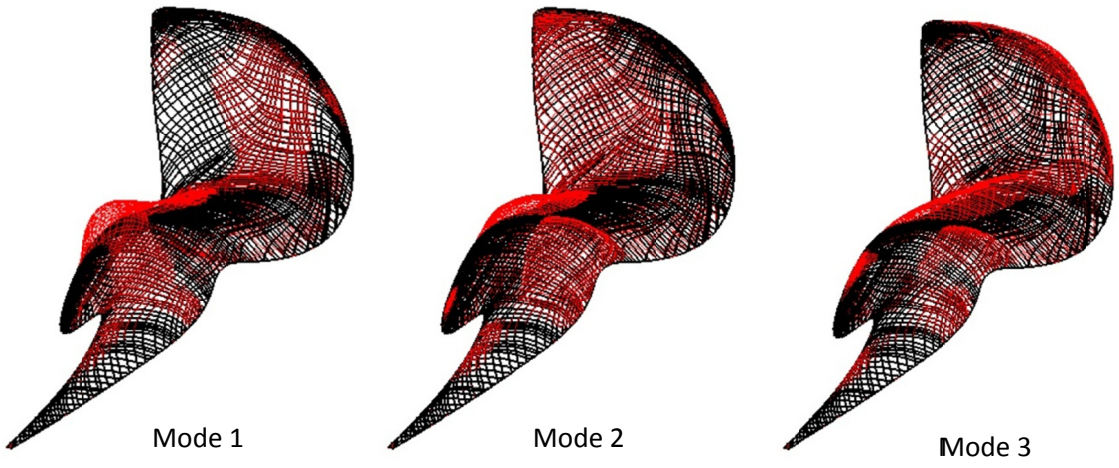


Figure 49 - The three fundamental mode shapes for the principal stress grid

5.3 Comparing the results

The returned results need to be compared with the structural performance of grids generated by conventional techniques. For this, three different grids were generated using the following methods:

(1) Slicing the surface using an equal grid of extending planes
(2) Following the NURBS surfaces' isoparametric lines using different parameters to create two different grid layouts

Parametric models utilizing each method were generated in Grasshopper and were automatically sent for analysis in Robot Structural.

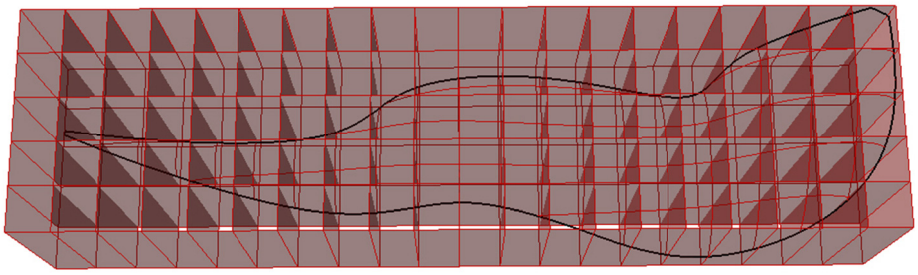


Figure 50 - Slicing planes for equal grid generation

To aid the results' coherence and enable comparisons between them, the mass of the structure for all three different cases should remain constant. For that reason, the total length of the trajectories of the Principal Stress grid was used as an unvarying parameter. Grasshopper was used in this context, enabling parameter change of each different model to achieve the highest similarity of trajectory lengths. The three comparison models with approximately equal mass are illustrated in Figure 52. The first one was generated by slicing the surface with equally spaced planes until reaching the desired length of grid lines (see Figure 50), the second and third were generated by laying grid rods along the surfaces isoparametric curves created by a U-V parameterisation, the latter being in addition refined for equal grid cell dimensions (see Figure 51).

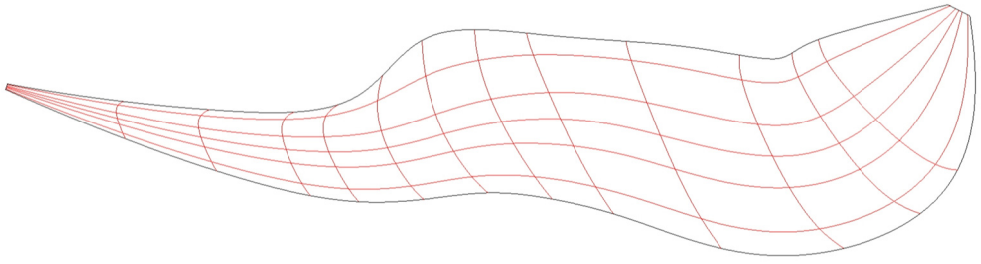


Figure 51 – Isoparametric lines of the surface

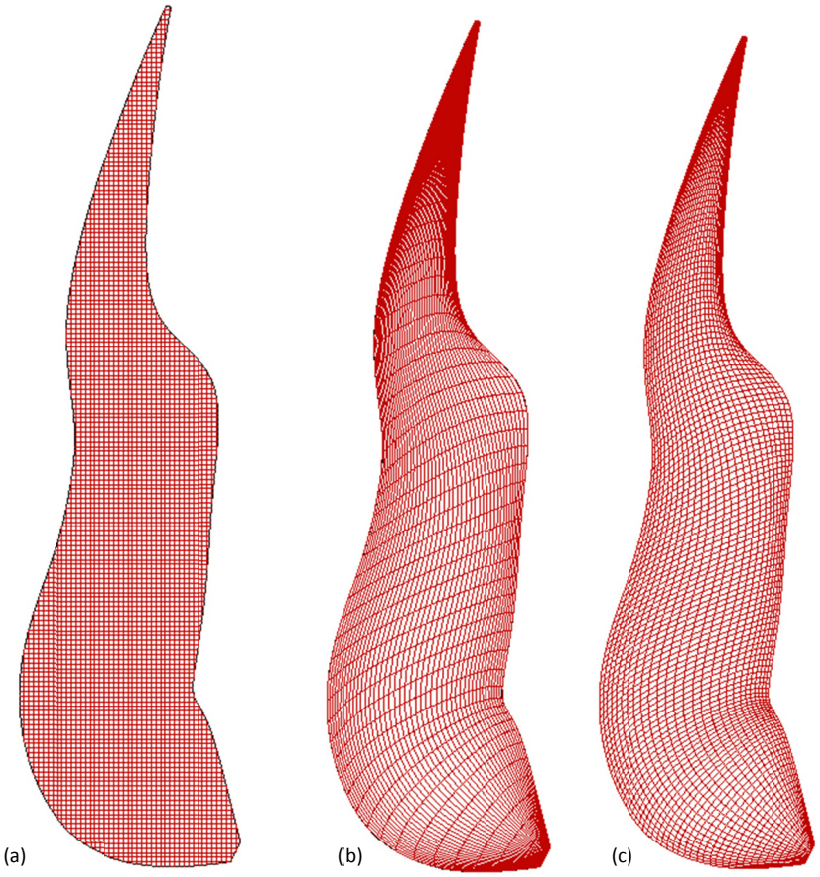


Figure 52 - The three different grids generated using conventional techniques. (a) Sliced surface, (b) Isoparametric 1, (c) Isoparametric 2

All three cases were analysed in Robot Structural Analysis using both a static and modal analysis. The structural analysis results for maximum deflections under self-weight, SLS load case and natural frequencies (taking into account the structures' self-weight), are summarized in the table below:

Model Type	Mode No.	Frequency(Hz)	Time Period(sec)	Total Mass(kg)	Self-weight deflection (mm)	SLS Deflection (mm)
Sliced surface	1	3.55	0.28	26187.80	15.93	24.14
	2	4.17	0.24			
	3	4.52	0.22			
Isoparametric 1 (U=44,V=49)	1	1.28	0.78	25891.35	45.45	119.62
	2	1.43	0.70			
	3	1.79	0.56			
Isoparametric 2 (Max Edge Length = 0.82)	1	2.74	0.36	26170.12	8.93	33.11
	2	3.06	0.33			
	3	3.27	0.31			

Table 2 – Analysis results for conventionally designed grids

The analysis results for the grids generated with conventional methods show significantly lower performance than the principal stress grid. Specifically the grid following the principal stress trajectories is much stiffer, having a natural frequency of 4.39 Hz, than the one created by slicing the surface with an equal sized grid which measured 3.55Hz in its first fundamental frequency. The grid following the isoparametric curves (with maximum edge length of 0.82) measured 2.74 Hz, which is significantly lower than both the principal stress grid and the sliced grid. The analysis of the grid generated following the isoparametric curves by defining U and V coordinates has revealed a very low stiffness caused by the uneven distribution of rods (elongated cells). However, not a realistic structure it was worth including this as its easily explicable flexibility increases confidence in the method of analysis. In addition, as shown in Table 1, the principal stress grid appears to deflect considerably less both under its self-weight (5.27mm) and under SLS load case (14.14mm) than the other three counterparts (Table 2).

The above results prove that the principal stress grid is considerably more efficient in minimising deflections under both the action of self-weight and imposed loads (see Figure 53). The modal analysis confirms this, since it shows that the principal stress grid is significantly stiffer than its counterparts created using conventional techniques.

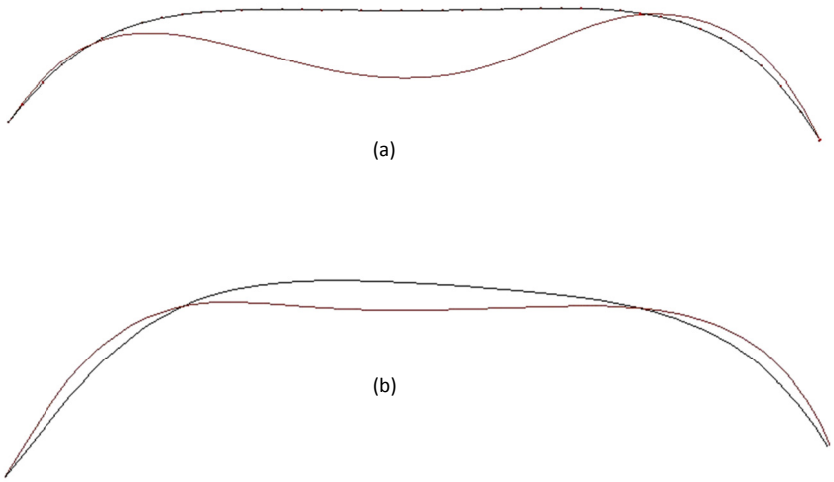


Figure 53 - (a) maximum deflection of the Sliced Surface grid, (b) maximum deflection of the Principal Stress grid - Sections taken at the points of global maximum deflections.

5.4 Optimisation of section sizes

Up to this point, the principal stress plotting algorithm was not returning a measure of the principal stress distribution along the surface. In order to optimise the sizes of the rods in the generated grid, the algorithm needed to be revised to include a stress value output rather than only stress direction output. The structural analysis component was outputting the stress values initially, however these were not included in the plotting process. To accomplish this, at every plotting step (at every successful intersection action), the value of the principal stress belonging to each stress direction involved was stored in a new data list.

The *polylines* were then plotted by using Rhino's programming interface rather than using Grasshopper components, so that their indexes coincided with the data in the stress values list. After a successful run of the plotting algorithm, the output included trajectories also associated with their inherent stress values. By using the Grasshopper's *colour gradient* component these results were visualised in Rhino. Their association with principal stress maps output for both directions is visualised in the set of images below (Figures 54-56).

Figure 54 - Principal stress ratios on the plotted trajectories

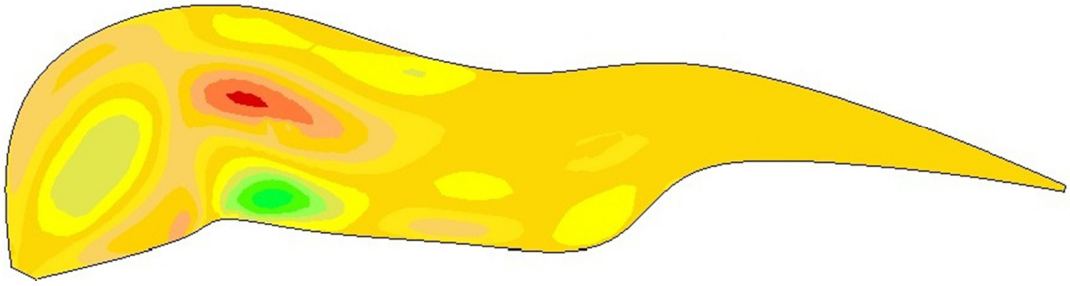


Figure 55 – Direction 1 Principal stress maps

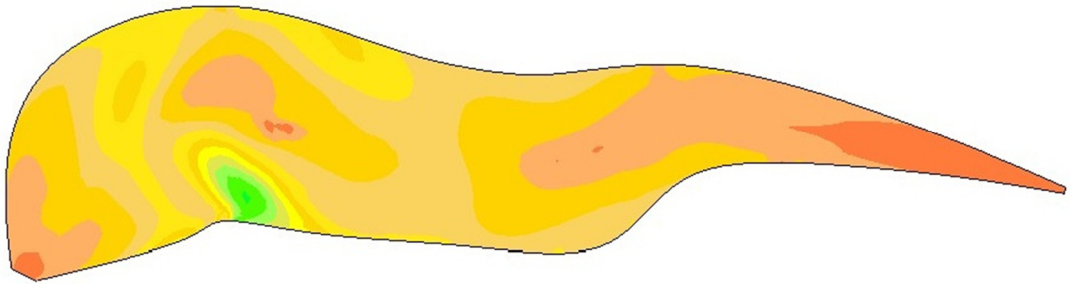


Figure 56 – Direction 2 Principal stress maps

The data generated aided the optimisation of the section dimensions of each rod according to the stress it is subject to. The stress values were then combined in grasshopper to create surface extrusions by giving a different depth to each plotted trajectory relative to the stress associated with it.

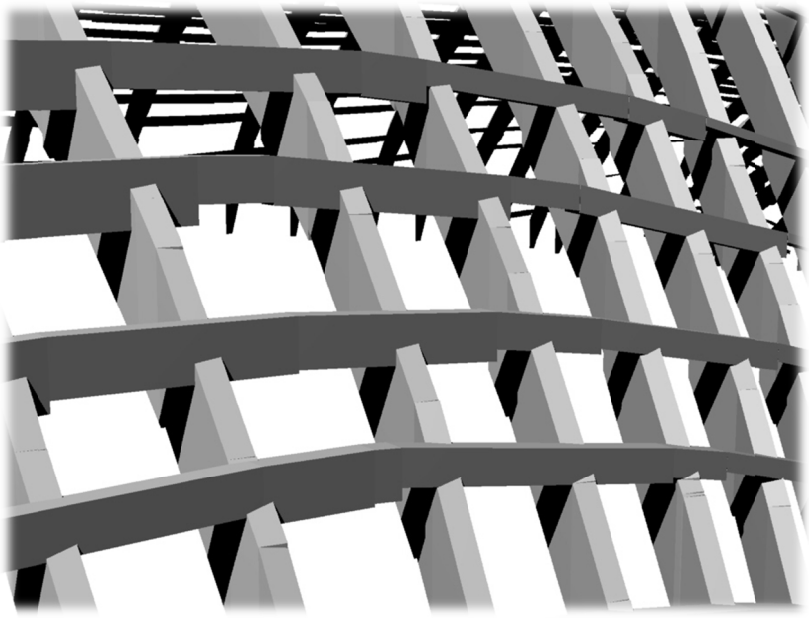


Figure 57 - Rendering - Tapered sections

The renderings that follow illustrate the resulting three-dimensional structure created using the principal stress plotting algorithm. The rods in the following cases have uniform depth sections.

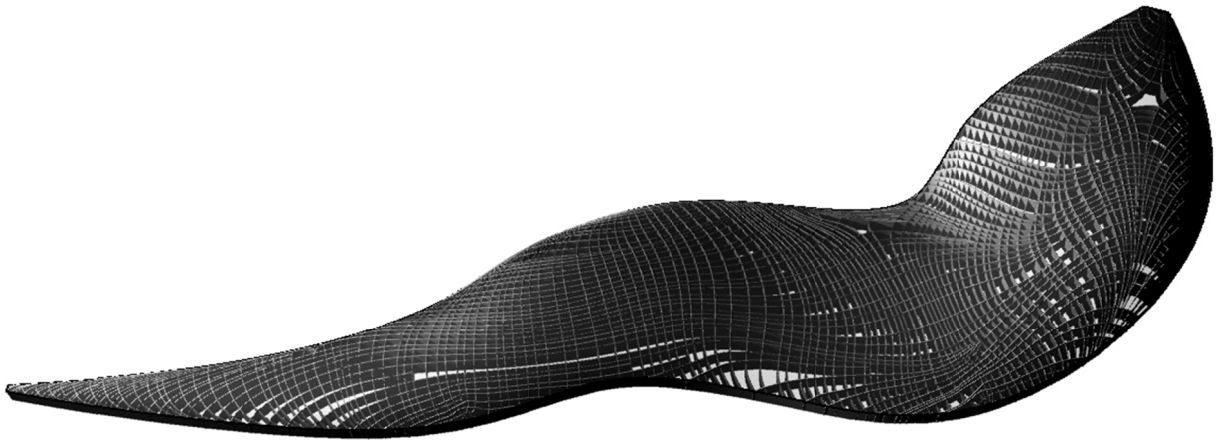


Figure 58 - Rendering perspective of the roof structure

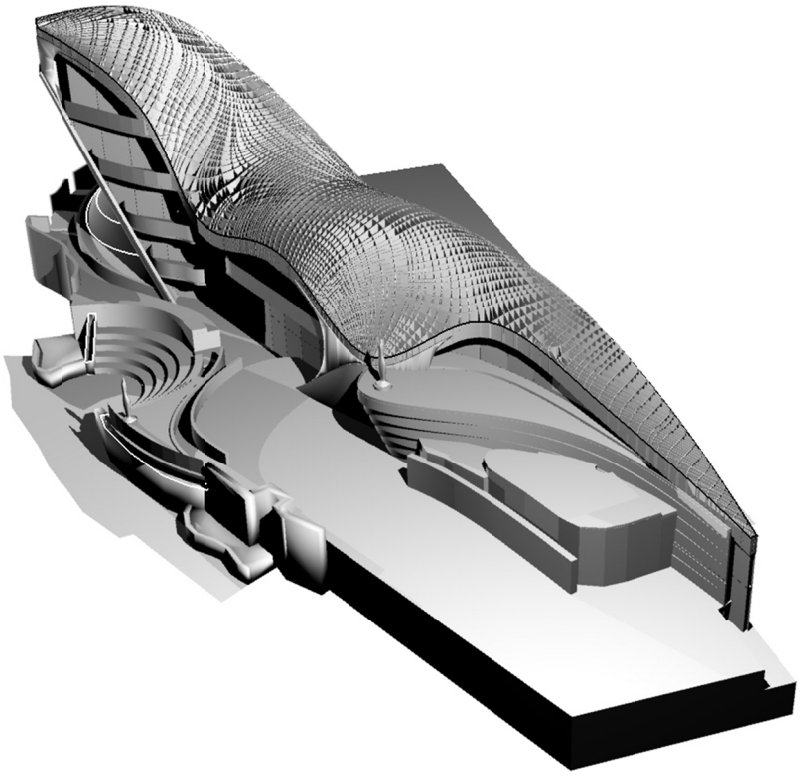


Figure 59 - Rendering perspective of the building and surroundings

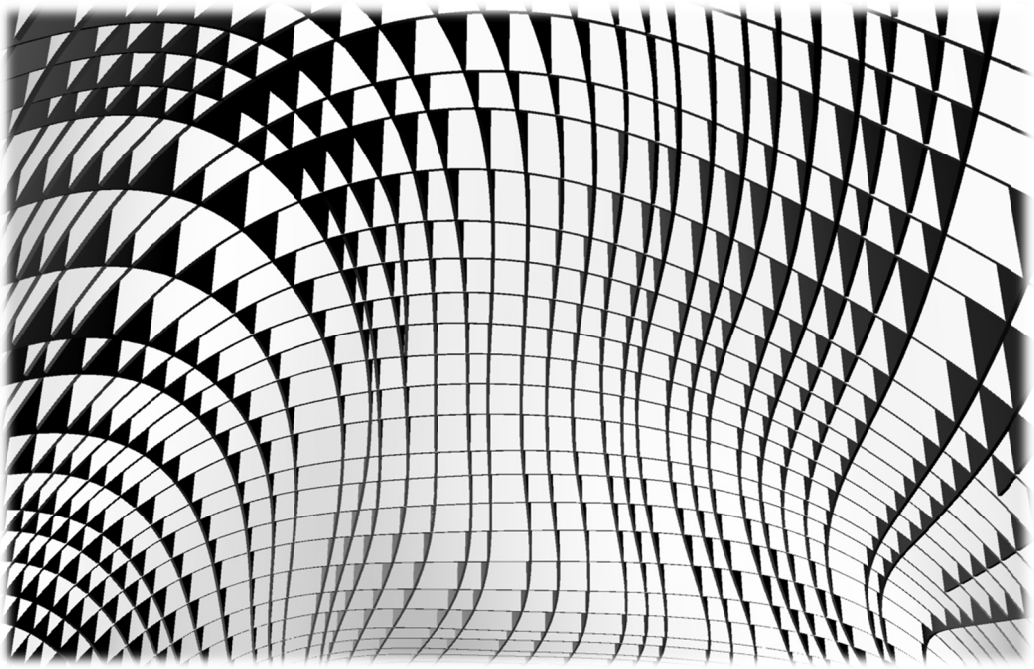


Figure 60 - Rendering - interior perspective of the roof structure

6 Conclusions

6.1 Overview

This research aimed initially at creating tools to achieve design interoperability between architecture and structural engineering, leading to the creation of efficient structures for complex architectural shapes. This was accomplished by employing the power of parametric design combined with structural engineering software. What is more, this combination enabled an interactive approach to structural design, a function which currently is sparingly applied for solving engineering problems. As documented in the literature review, tools that aid fast and associative design are readily available to architects. These allow them to generate, analyse and manipulate models quickly, while being able to visualise results in a real-time manner. This process often leads to architecturally pleasing design solutions. The involvement of a structural engineer in the early stages of design is essential in modern architecture and particularly in regards to complex forms. Therefore, without the use of compatible tools, an engineer not only fails to support architectural design but also moves away from efficient solutions. Although there is still a large field of applications to explore and exploit, the following were successfully developed during this research:

Parametric definition of a 2D truss

A two-dimensional truss was fully described in a parametric manner in Grasshopper 3D, by including both its geometric and structural attributes. The attributes included its span, its height, the bar section sizes, supports position and type, the type of truss (Brown, Pratt or Vierendeel) and load case or load combination selection. All the attributes by which it was described could be manipulated in Grasshopper by either using numerical sliders or text panels for assigning string inputs. The latter were used for the cases of selecting attributes through already defined lists as for example section types. The truss was analysed by transmitting the data to Robot Structural Analysis which returned the desired results to Grasshopper's (Rhino 3D) environment. For any change of attributes, the truss would interactively update and visualise the new results output.

Interactive analysis of a free form surface

This application enables any free form NURBS surface to be translated into an analytical surface for an FE analysis. In a similar manner as with the 2D truss, all the surfaces attributes (geometrical and structural) were defined parametrically in Grasshopper. These attributes included the geometric definition of the surface and its mesh geometry, the support constraints such as column supports and their position, or surface perimeter supports, as well as material and thickness of the shell. Again, the surface was sent to RSA for analysis and the results were returned to Grasshopper's environment interactively, being given any possible change of the former or its structural attributes.

Parametric truss form-finding

In an attempt to further exploit the potential of the parametric truss definition, the previous 2D truss example was extended in order to investigate more efficient forms of trusses. By utilising the analysis output, the geometry of the truss was reshaped following the force diagrams of its members to lead to a more even distribution of stresses. This method produced some interesting novel shapes of trusses which fully correspond to the internal stress conditions of its members and thus minimise the material in the areas where it is not utilised.

Developing structurally efficient grid structures

The major part of this research concerned the generation of efficient structures to support free form architectural designs. This was achieved by making use of the experience gained from the previous, rather simple examples, relating to a variety of problems extending from interoperability to form finding. A technique of utilising the areas of maximum and minimum stress that occur in a continuum was used. Those areas were isolated in a form of grid which efficiently substituted the continuous free form surface. For this purpose, the framework for acquiring analysis data from structural analysis software that was created in Section 3.2 was used. The directions of the principal stresses emerging from the analysis of the surface (assuming a continuous concrete shell) were used as the guide to plot trajectories that would later produce the proclaimed grid. The efficiency of the generated structure was proved by comparing its performance with grids created using conventional techniques for a real design project. The principal stress grid for the tested surface yielded significantly increased stiffness compared to equal weight conventional grids. In addition, it appeared to deflect much less than both conventional grids.

6.2 Further Work

This thesis has demonstrated the efficient development and application of tools that aid interoperability in design and in parallel contribute to the generation of efficient structures. Although there is still a large field to be investigated and explored by engineers ,this research has pioneered the generation of a variety of interactive applications to provide the designer with significant assistance, ranging from solutions to small engineering problems to complex programming-intensive applications aiding free form design. What is more, it will hopefully trigger further invention in the field, as it has shown the potential for ample exploitation.

Some further improvements to the applications which were generated during this research should be however highlighted in order to be realized in future work:

Initialisation of the Robot Structural Analysis member design engine

The utilisation of this interface and its linking with Grasshopper 3D will lead to the improvement of the performance of the created structures. The interfaces' inbuilt functions would allow designing and checking members under all possible design codes while

ensuring their structural sufficiency. The structures generated in both the case of the truss and the free form surface grid were not designed nor checked in their ability to withstand 2nd order effects as buckling. The further extension of the tools to include and control the *member design interface* would therefore enable the assessment of such structural behaviour and the anticipated rigorous design of the structure's components. Furthermore, the utilisation of the *plate and shell reinforcement design* interface for would enable the realisation of concrete reinforcement layouts on any free form surface, making use relevant tool created in this context.

Improving the trajectories plotting algorithm

One of the major issues that remained unresolved regarding the plotting process was the sudden termination of trajectories at the points were the algorithm failed to choose the correct direction for the next plotting step. The error was inevitable since the continuum problem was translated to a discretized one and the algorithm was forced to select a direction associated with each facet. The fault occurred when the selected direction was not co-planar with the next in sequence face (see Section 4.7*)*. This error could possibly be administered if in every termination of the plotting routine, a new trajectory was initiated, having the same direction, but starting from a different point adjacent to the point of termination within the associated face. The achievability of this method was partly illustrated in the *Random selection process,* where each trajectory would restart after termination to ensure that both opposing vector directions were initiated from each face.

Another part of the plotting algorithm that requires further development is the spacing control of the trajectories. In the current context, two methods were used: a manual method of re-plotting each trajectory by its starting point by using numerical sliders and a semi-automatic method of stitching together trajectories that are close to each other. However, none of the two approaches is embedded in the actual plotting algorithm but they are applied in a post processing stage to refine the produced result. This raises the need for the development of an inherent routine that controls the trajectory spacing through a more systematic approach.

Furthermore, the implementation of techniques and algorithms that were applied successfully in resampling surface meshes (for use in the animation industry) as "Periodic global parametrisation" [30] and "Spectral Surface Quadrangulation" [34] could initiate a new approach to the problem of plotting trajectories that follow principal stresses occurring in a continuum shell.

References

1. Veltkamp,M., 2007. Freeform Structural Design. Thesis (PhD). Delft University Of Technology.
2. Future Systems [online]. Available from : http://www.future-systems.com/architecture/architecture_03.html [Accessed 01 July 2010].
3. Modelling tools for designers [online]. Available from : http://www.rhino3D.com/cad.htm [Accessed 01 July 2010].
4. Rhinoceros- What is NURBS? [online]. Available from : http://www.rhino3D.com/nurbs.htm [Accessed 01 July 2010].
5. Grasshopper- Generative modelling for Rhino [online]. Available from : http://www.grasshopper3D.com/ [Accessed 25 June 2010].
6. Maya 3D Animation, Visual Effects and Compositing Software – Autodesk [online]. Available from : http://usa.autodesk.com/adsk/servlet/pc/index?siteID=123112&id=13577897 [Accessed 03 July 2010].
7. BIM and Beyond – Generative Design: Generative Components [online] . Available from: http://www.bentley.com/en-US/Products/GenerativeComponents/ [Accessed 03 July 2010].
8. CATIA – PLM Solutions – 3D Systems Shape Mechanical and Equipment Virtual Design – Dassault Sys. [online]. Available from: http://www.3Ds.com/products/catia/ [Accessed 01 July 2010].
9. Wikipedia, Pantheon, Rome. In: Wikipedia: the free encyclopedia[online]. St Petersburg, Florida: Wikimedia Foundation. Available from: http://en.wikipedia.org/wiki/Pantheon,_Rome#Structure [Accessed 22 June 2010].
10. Eduardo Torroja Institute[online]. Available from: http://www.ietcc.csic.es/, [Accessed 06 July 2010].
11. Candela Projects, A Project of Felix Candela and the Urban Condition of Mexico City [online]. Available from : http://www.candelaprojects.com/ [Accessed on 06 July 2010].
12. Nervi, P.L.,1956. Structures. USA : F.W. Dodge Corporation
13. Mainstone, R.J.,1998. Developments in Structural Form. 2nd ed. Oxford : Architectural Press.

14. Aish, R.,Woodbury,R., Multi-Level Interaction in Parametric Design. Bentley Systems Incorporated.

15. Papalexopoulos D., Stavridou A., Papadopoulos D., 2007, Conceptual determination of parametric attributes of architectural construction elements and materials (free translation) . National Technical University of Athens, Athens.

16. Aish. R., Bentley's GenerativeComponents-A design tool for exploratory architecture . Bentley Publications.

17. 610:1990. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. Institute of Electrical and Electronics Engineers.

18. Georgiou, O., 2009. MediaCite- Design to Fabrication. Research Report, University of Bath.

19. Autodesk- Autodesk Robot Structural Analysis Professional[online]. Available from:http://usa.autodesk.com/adsk/servlet/pc/index?siteID=123112&id=11818169 [Accessed on 20 June 2010].

20. Wikipedia, Application Programming interface. In: Wikipedia: the free encyclopedia[online]. St Petersburg, Florida: Wikimedia Foundation. Available from: http://en.wikipedia.org/wiki/Application_programming_interface , [Accessed 2/07/2010].

21. Autodesk Inc., 2009. Autodesk Robot Structural Analysis 2010 – Robot Object Model.

22. O. C. Zienkiewicz, R. L. Taylor, Robert Leroy Taylor, J. Z. Zhu, 2005. The finite element method: its basis and fundamentals-The standard discrete system and origins of the finite element method. 6th ed. Elsevier. Available online from : http://knovel.com/web/portal/browse/display?_EXT_KNOVEL_DISPLAY_bookid=1886&VerticalID=0 [Accessed on 15 June 2010].

23. Mesh [online]. Available from: http://www.rhino3D.com/4/help/Dialogs/DocumentProperties_Mesh.htm [Accessed 31 August 2010].

24. Developer: dotnetplugins- McNeel Wiki [online] Availble from: http://wiki.mcneel.com/developer/dotnetplugins [Accessed 20 July 2010].

25. Wikipedia, I-beam In: Wikipedia: the free encyclopedia[online]. St Petersburg, Florida: Wikimedia Foundation. Available from: http://en.wikipedia.org/wiki/I-beam [Accessed 18 July 2010].

26. Littlefield D., ed., 2008. Space Craft-Developments in Architectural Computing-Geometry form and complexity. London: RIBA Publishing.

27. Massimiliano Fuksas Architetto/ Progetti/ New Trade Fail Milano [online]. Available from : http://www.fuksas.it/#/progetti/0703/ [Accessed 16 July 2010].

28. Winslow, P., 2009.Synthesis and Optimisation of Free-Form Grid Structures. Thesis (PhD). University of Cambridge .

29. Michalatos, P., and Kaijima, S. Design in a non homogeneous and anisotropic space. Symposium of the International Association of Shell and Spatial Structures, Venice, Italy, December 3-6 (2007).

30. Ray, N., Li, W., Levy, B., Sheffer, A. and Alliez, P. Periodic global parameterization. ACM Transactions on Graphics, 25(4) (2006), pp. 1460 - 1485.

31. Kotsovos,M.D.,2005. Reiforced Concrete. Athens: NTUA publications .

32. Wikipedia, Umbilical Point In: Wikipedia: the free encyclopedia [online]. St Petersburg, Florida: Wikimedia Foundation. Available from: http://en.wikipedia.org/wiki/Umbilical_point [Accessed 23 July 2010].

33. Line – Line Intersection from Wolfram Mathworld[online]. Available from: http://mathworld.wolfram.com/Line-LineIntersection.html [Accessed on 21 July 2010].

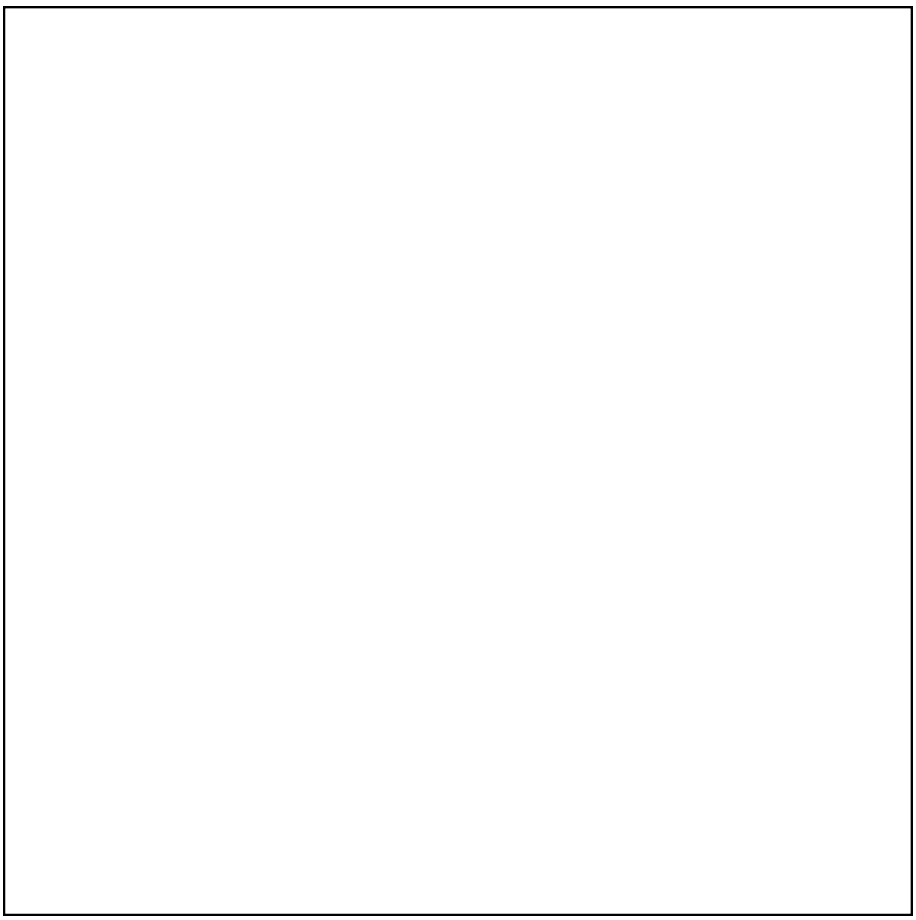
34. S. Dong, P-T. Bremer, M. Garland, V. Pascucci, and J. C. Hart. Spectral surface quadrangulation. ACM Transactions on Graphics, Proceedings of SIGGRAPH 2006

35. Kristina Shea, Robert Aish & Marina Gourtovaia. (2005).Towards integrated performance-driven generative design tools. Automation in Construction. March 2005, 14(2).pp. 253-264.

36. Wikipedia, Industry Foundation Classes. In: Wikipedia: the free encyclopedia [online]. St Petersburg, Florida: Wikimedia Foundation. Available from: http://en.wikipedia.org/wiki/Industry_Foundation_Classes [Accessed 04 October 2010].

A. Appendix

A1. DVD

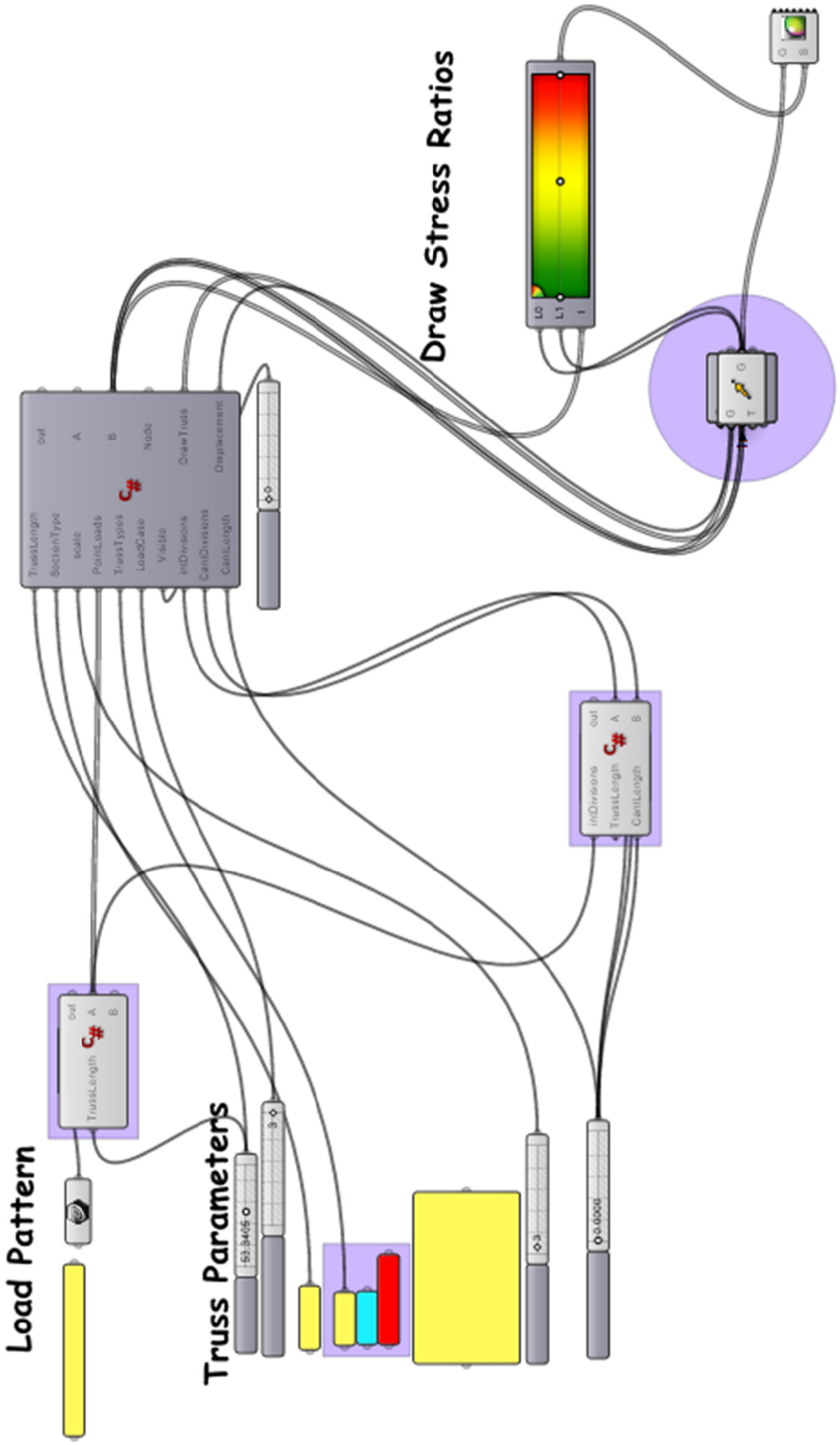


DVD Contents

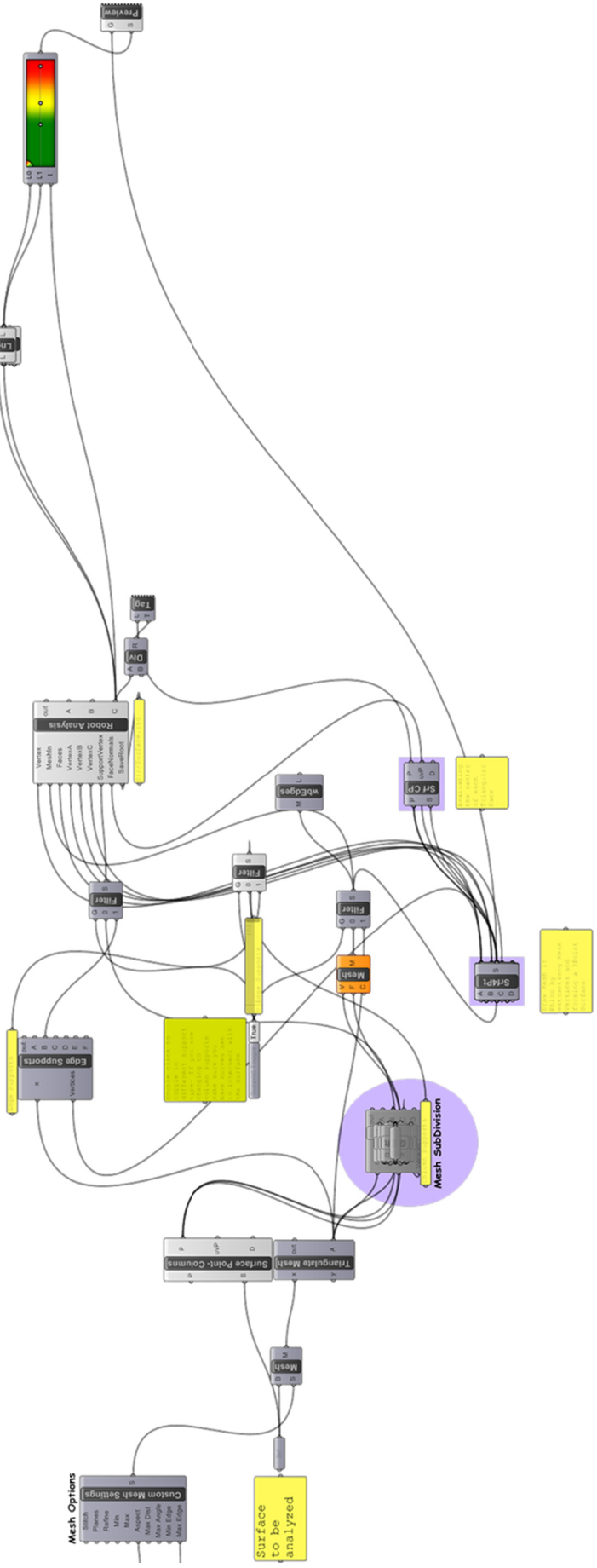
- Developing structurally efficient grid structures
  - ComparativeTests
    - GH scripts
    - Rhino models
    - RobotModels
  - PrincipalCurvutures
  - Rhino Models
  - General Rhino Models used
  - Interactive analysis of free form surface
  - Parametric Definition of 2d Truss
  - Parametric Truss form-finding
  - Referenced Assembly
  - Thesis Images
    - Renders
  - Written documents

A2. Grasshopper Models (screenshots)

Parametric Truss Definition and form-finding of a 2D truss



Interactive analysis of a free form surface



Developing structurally efficient grid structures